IMPLEMENTACION DE UNA MALLA COMPUTACIONAL EXPERIMENTAL EN LA UNIVERSIDAD AUTONOMA DE OCCIDENTE.

JAIRO FERNANDO ORTIZ ZAPATA

UNIVERSIDAD AUTONOMA DE OCCIDENTE
FACULTAD DE INGENIERIA
DEPARTAMENTO DE CIENCIAS DE LA INFORMACION
PROGRAMA DE INGENIERIA INFORMATICA
SANTIAGO DE CALI
2008

IMPLEMENTACION DE UNA MALLA COMPUTACIONAL EXPERIMENTAL EN LA UNIVERSIDAD AUTONOMA DE OCCIDENTE.

JAIRO FERNANDO ORTIZ ZAPATA

Trabajo de grado para optar por el titulo de
Ingeniero en Informática

Director
ALEXANDER GARCIA DAVALOS
Ingeniero de Sistemas

UNIVERSIDAD AUTONOMA DE OCCIDENTE
FACULTAD DE INGENIERIA
DEPARTAMENTO DE CIENCIAS DE LA INFORMACION
PROGRAMA DE INGENIERIA INFORMATICA
SANTIAGO DE CALI
2008

Nota de aceptación:

Aprobado por el Comité de Grado en Cumplimiento de los requisitos Exigidos por la Universidad Autónoma de Occidente para optar al Titulo de Ingeniero de la Informática

OSCAR MONDRAGÓN

Jurado

MIGUEL NAVAS

Jurado

Santiago de Cali, 04 de junio de 2008

Este trabajo de grado esta dedicado en un 90% a Dios y el otro 10% a mi familia y a una persona que me concluyó a pensar que a partir del ser humano se escribe una realidad que solo las personas más importantes para ese ser pueden ver.

AGRADECIMIENTOS

Agradezco a todas las personas que hicieron este trabajo posible, a mi familia por el apoyo constante e incondicional, a mi director por guiar siempre el proyecto por su rumbo, a las personas de la Universidad de los Andes (Harold, Eliza, Andrés, Michael, Mauricio) y definitivamente a todos los que anteponen el estudio por encima del dinero temporal.

CONTENIDO

LISTA DE TABLAS

Pág.

LISTA DE FIGURAS

Pág.

LISTA DE ANEXOS

RESUMEN

El siguiente trabajo consta de una implementación de una nueva tecnología que permite el aprovechamiento de poder de cómputo sub-utilizado para la ejecución de algoritmos que exijan gran poder computacional. Para este trabajo se implementa un cluster, el cual es la base de un Grid y a partir de varios clusters se diseña un Grid. El cluster presenta diferentes recursos computacionales pero pertenecen al mismo dominio, sin embargo la tecnología Grid permite la interacción de recursos computacionales de varios clusters mediante lo que se conoce como Virtualización y aprovisionamiento.

La ubicación descentralizada de los algoritmos que se van a ejecutar en el Grid es uno de los aspectos mas importantes, ya que una de las características más relevantes del Grid es que en este se comparten recursos más no los algoritmos como tal.

Adicionalmente, se mostrará cuán importante es la investigación y los principales proyectos a nivel mundial y a nivel nacional que se están desarrollando bajo esta tecnología, mencionando también las personas encargadas y como se puede llegar a ser parte de un proyecto a nivel mundial.

Este trabajo, además, contiene un caso de estudio el cual permite ver el rendimiento de un algoritmo de vuelta atrás, el cual requiere de un gran nivel de procesamiento de datos, bajo dos ambientes diferentes: Computación en Grid y computación tradicional. También tiene otro caso de estudio en el cual se implementa el algoritmo de QuickSort para un gran volumen de datos, el cual permitirá observar la diferencia de los tiempos de ejecución de una gran cantidad de datos procesados bajo un ambiente tradicional contra un ambiente en Grid.

INTRODUCCION

En el presente trabajo se presenta un tema de actualidad, con el cual la Universidad Autónoma de Occidente se verá beneficiada en el mejoramiento del procesamiento de grandes cantidades de datos y el procesamiento en paralelo.

Este trabajo trata de la implementación de una Malla Computacional, la cual se caracteriza por mejorar los tiempos de procesamiento de información mediante la utilización de recursos ociosos de los recursos computacionales que hagan parte de una red.

Grandes empresas en la actualidad están implementando esta nueva arquitectura computacional ya que los beneficios que implica la mejora en sus sistemas de información van más allá del mejoramiento o rendimiento de los mismos. Además, Grid Computing ofrece un menor costo, reduce la inversión, no es necesario invertir o rearmar una nueva red, entre muchos otros beneficios.

En Colombia, la investigación en el tema de Grid Computing está liderada por la Universidad de los Andes, con su proyecto Campus Grid mediante la implementación de la tecnología Globus Toolkit en los computadores de la Universidad.

La finalidad del presente trabajo consiste en experimentar con los beneficios que ofrece la Malla Computacional en los laboratorios de la Universidad Autónoma de Occidente y de que manera se puede usar como herramienta de apoyo a la investigación en la Institución.

1. PLANTEAMIENTO DEL PROBLEMA

La Universidad Autónoma de Occidente, en el departamento de Ciencias de la Información ha planteado problemas relacionados con la optimización de recursos computacionales que hacen parte de la red de la Universidad.

El primero de estos problemas va relacionado con ¿Cómo utilizar de manera adecuada los recursos inutilizados de los computadores y demás recursos que hacen parte de la red con el fin de que se puedan computar tareas que requieran un alto rendimiento en procesamiento de datos? Esta pregunta va relacionada a los diferentes problemas que se presentan en grupos de investigación como mecánica de fluidos y tratamiento de imágenes cuando se requiere procesar grandes cantidades de datos, lo que demanda el poder computacional de una gran computadora y otros recursos computacionales bastante costosos. Grid Computing o Malla Computacional es una alternativa interesante para ser analizada y dependiendo de los resultados, implementada o ignorada.

Ya teniendo identificada la primera pregunta, el segundo problema aparece de la siguiente manera: ¿Qué herramienta utilizar para implementar Grid Computing en la Universidad Autónoma? Esta pregunta es un poco más compleja de responder ya que la implementación de Grid Computing en nuestro país es relativamente nueva y recientemente se crearon proyectos como Grid Colombia, el cual tiene como objetivo la difusión a través de la realización de conferencias de diferentes entes institucionales a nivel nacional y la cooperación en proyectos de ingeniería.

Finalmente, definir, mediante la comparación de los tiempos de ejecución de un algoritmo en dos ambientes diferentes (Computación en Grid y Computación Tradicional), los beneficios para la organización que la efectúe.

2. ANTECEDENTES

Los antecedentes van ligados con la implementación de herramientas que permitan al Grid Computing procesar grandes volúmenes de información en diferentes proyectos de carácter científico, comercial, entre otros.

Actualmente vale la pena mencionar, el proyecto de la comunidad científica denominado "SETI@home"[1] es el mayor proyecto de computación distribuida en existencia. También puede ser considerada como la supercomputadora más grande existente y el cálculo computacional más grande de la historia, ya que este proyecto surgió de la necesidad de investigar la existencia de vida extraterrestre mediante telescopios y ondas radiales, proyecto realizado por SETI (Search for Extraterrestrial Intelligence at Home), Al hacer esta búsqueda en otros planetas era imposible utilizar un telescopio para identificar vida en otro lugar, por más grande que fuera no tenía alcance, entonces se decidió capturar la emisión de ondas de radio. Para esto se instaló en Arecibo, Puerto Rico, un radiotelescopio con un diámetro de 305 metros, el más grande del mundo, el cual por su gran potencial captura hasta las señales más débiles en el espacio. Sin embargo para que un solo computador procesara la información recogida por el radiotelescopio, tomaría muchísimo tiempo y además se necesitaría un superordenador muy costoso. En vista de este inconveniente el proyecto decide utilizar la ayuda pública, que consiste en que personas interesadas en el proyecto donen los tiempos muertos de sus computadores para que en este tiempo SETI@home se active y pueda trabajar y procesar los datos adquiridos en búsqueda de inteligencia artificial.

Con este fin se creó un Grid que permite compartir los recursos infrautilizados que posean los computadores pertenecientes al Grid, independientemente que posean diferentes arquitecturas, programas, sistemas operativos o ubicación geográfica. Así se logró tener conectadas todas las personas del proyecto y de esta manera utilizar varios equipos para agilizar el tiempo de procesamiento de los datos y ahorrar en la compra de un superordenador.

Las señales recogidas por el radiotelescopio son enviadas a la Universidad de Berkeley en California, donde se encargan de dividir la información capturada en pequeños paquetes que son posteriormente enviados a cada uno de los miembros del Grid para que éstos los procesen. Luego de esto, los datos son enviados de vuelta a la Universidad de Berkeley donde los resultados son ordenados y estudiados por categorías según su orden de importancia.En el campo de la medicina, el Dr. Robert Hollebeek, director del National Scalable Cluster Lab en la Universidad de Pensilvania en Philadelphia, en colaboración con un

[1]Inteligencia Extraterrestre [en linea]: SETI. California: SETI@HOME, 2007. [Consultado el 23 de octubre de 2007]. Disponible en Internet: http://setiathome.ssl.berkeley.edu

Grupo de Oak Ridge National Laboratory, se está implementando un Grid que va a albergar toda la información de diagnósticos y tratamientos realizados a pacientes con cáncer de mamas de manera que los resultados de las mamografías se podrán comparar con imágenes digitales de mamografías anteriores y así detectar de forma temprana la presencia de cualquier inconsistencia en el paciente. Además el Grid proveerá a sus usuarios de herramientas especializadas para el análisis de las muestras. Cada hospital tendrá un portal con dos servidores en los cuales se van a almacenar repositorios de imágenes digitales e información para compartir con el resto del Grid.

Otro proyecto importante basado en Grid se lleva a cabo actualmente en La Universidad Nacional de Colombia dirigido por Gabriel Mañana, Germán Hernández, Luís Fernando Niño, Oscar Agudelo y Erika Ardila. El proyecto Ungrid busca poder almacenar radiografías de pacientes durante los tratamientos y comparar de forma paralela las recientes con anteriores para observar su progreso. Para esto necesitan contar con un computador virtual que tenga mucha capacidad de almacenamiento y procesamiento, esto se logra, tomando los tiempos muertos de los computadores con los que cuenta la Universidad Nacional en su sede de Bogotá. Para la realización del proyecto Ungrid se utilizó una tecnología llamada "JavaSpaces"[2], desarrollada por Sun Microsystems, desarrollado en lenguaje Java. Java se caracteriza por ser multi-plataforma, lo cual es un beneficio en el desarrollo de la computación Grid a partir de una plataforma heterogénea.

Observando los beneficios que brinda esta nueva tecnología, empresas reconocidas como IBM, Sun Microsystems, Intel y HP se han unido al gran potencial que les representa la Computación en Malla. En Febrero de 2003; "IBM"[3], proveedor mundial líder de sistemas, y Butterfly.Net, proveedor líder de tecnología en red y servidores para juegos en línea; vieron que el funcionamiento tradicional de video juegos en línea era poco eficiente; ya que en el momento en que el servidor del juego se sobrecargaba no permitía que más jugadores entraran a éste, limitando de esta forma los participantes. Con la nueva tecnología Grid iniciaron el proyecto Butterfly.Net.Inc, que podía soportar más de un millón de jugadores simultáneamente, ya que esta tecnología provee recursos computacionales a las áreas más visitadas o congestionadas, brindando un rendimiento óptimo y eficiente. La arquitectura Grid permite que el servidor tanto de los juegos como de las bases de datos utilice fibras ópticas de alta velocidad habilitando la ruta de los usuarios a diferentes partes o servicios del Grid y garantizando disponibilidad, rendimiento y seguridad en la información. Para poder hacer uso de este servicio los jugadores deberán registrarse en una página (www.butterfly.net), luego de esto recibirán un kit con: un software para instalar en su computador, librerías, documentación y soporte técnico.

Butterfly.Net es uno de los más grandes proyectos creados en el área de los video juegos, Los desarrolladores y proveedores de juegos para PlayStation®2 pueden utilizar Butterfly

[2] Javaspaces [en línea]: Reinvent The Data Warehouse. USA : SUN MICROSYSTEMS, 2007. [consultado el 12 de noviembre de 2007]. Disponible en Internet: http://www.sun.com

[3] IBM Grid Solutions [en línea]: IBM Grid Computing. New York: IBM, 2008. [Consultado el 8 de enero de 2008]. Disponible en Internet: http://www.ibm.com/grid

Grid para acceder a recursos de informática bajo demanda, reducir costos y aprovechar mejor las propiedades de sus juegos online.

A nivel nacional se destaca el Campus Grid de la Universidad de los Andes, el cual mediante la implementación de la herramienta "Condor"[4]. a nivel de Cluster y la implementación de Globus Toolkit a un nivel superior, busca aprovechar el poder de cómputo de todos los recursos de cada dependencia para establecer una infraestructura única de investigación, siempre respetando las prioridades administrativas de cada dependencia.

[4] Condor High Throughput Computing [en línea]: The Goal of the condor. Wisconsin: Universidad de Wisconsin, 2008. [Consultado 17 de febrero de 2008]. Disponible en Internet : http://www.cs.wisc.edu/condor/

3. MARCO TEORICO

El término **Grid Computing** o Computación en Malla encierra muchas definiciones, nuevos paradigmas, foros y discusiones sobre el tema debido a que desde la creación del Internet, la comunidad ha pensado en crear una red en donde se pueda aprovechar los recursos computacionales subutilizados del hardware perteneciente a esta red.

Se ha observado que la arquitectura de Computación Distribuida ha tenido una serie de fases o generaciones que vale la pena mencionar (ver tabla 1); destacando Grid Computing como la nueva generación de la Computación Distribuida.

Tabla 1. Las cinco generaciones de la computación distribuida

Generación	Características.
1. (computación basada en computadores)	• Terminales brutas. • Un solo servidor. • Aplicaciones monolíticas
2. Acceso Remoto.	• Un solo cliente soportando las funciones de los terminales. • Un solo servidor.
3. Cliente/Servidor.	• El cliente soporta algunos procesos mediante interfaces. • Dos servidores.
4. Computación multitarea	• El cliente soporta algunos procesos mediante interfaces. • Más de dos servidores.
5. Grid Computing	• Entorno virtual donde todos los sistemas son considerados un conjunto de recursos compartidos. • Arquitectura SOA.

Fuente: IDC Grid Computing 2004 [en linea]: IDC Analyze the Future. Indonesia: IDC, 2004. [Consultado el 05 de marzo de 2008]. Disponible en Internet: www.oracle.com/corporate/analyst/reports/infrastructure/dbms/210558.pdf

La definición de **Grid Computing** no está plenamente establecida de manera estándar ya que los diferentes trabajos de las diferentes empresas u organizaciones que investigan el tema intentan definir de la mejor manera el término **Grid Computing**; entre las definiciones más destacadas en la comunidad científica se encuentran las siguientes:

"… Grid Computing es el conjunto de todos los recursos computacionales dentro de un solo conjunto de servicios compartidos para todas las necesidades computacionales de las empresas"[5].

También cabe mencionar la definición realizada por Carl Kessenlman e Ian Foster, dos de los más grandes investigadores de la Universidad de Chicago en el tema de Computación Distribuida:"… Una grilla computacional es una infraestructura de hardware y software que provee un acceso seguro, consistente, penetrante y barato a capacidades computacionales de alto rendimiento"[6].

En Colombia los principales actores de la investigación de Grid Computing, Harold Castro y Elizabeth Guardo, definen como principal objetivo de Grid Computing la siguiente idea:

El objetivo de un grid es que cada participante obtenga un beneficio superior al que podría obtener actuando de manera aislada. Esta filosofía está permeando la comunidad científica internacional y el pertenecer a redes de grid internacionales, aportando cada uno su infraestructura, se está volviendo un requisito para participar en esa comunidad.

Las demás definiciones encontradas en los artículos publicados mencionan la misma idea de compartir recursos computacionales que no son utilizados, entre múltiples organizaciones (múltiples ambientes), que permitan la realización de tareas de alta carga computacional, tales como procesamiento intensivo de datos, supercomputación distribuida, simulación en 3D, entre otras aplicaciones.

Roberth G. Shimp, define dos aspectos fundamentales en la definición de Grid Computing, los cuales dice que deben hacer parte de las aplicaciones que implementen el paradigma de Grid Computing; estos aspectos son:

- **La Virtualización**, en donde precisa que este concepto va ligado a recursos computacionales que están unidos de manera abstracta y de manera virtual de modo que hacen parte de una red, cuyos componentes y/o recursos serán compartidos mutuamente; por ejemplo dos computadores ubicados geográficamente alejados y no relacionados de manera directa pueden ser parte de una red virtual mediante un sistema de información que permita, mediante protocolos de comunicación, compartir recursos entre ellos de manera confiable, penetrante, y segura.

[5] NASH, Miranda. Oracle 10g [en linea]: Infraestrucure for Grid Computing. New York: Oracle Corporation, 2003. [Consultado 15 de octubre de 2007]. Disponible en internet: http://www.oracle.com

[6] FOSTER, Ian. What is the Grid? [en linea]: A three point checklist. Chicago: University of Chicago, 2002. [Consultado 20 de octubre de 2007]. Disponible en internet: http://www.mellanox.com/pdf/whitepapers/GridTech_WhitePaper_final.pdf

- El **Aprovisionamiento** de la información, en el que menciona, es la identificación plena de los requerimientos enviados por el usuario con el fin de cumplirlos a cabalidad, direccionando las tareas al recurso más apropiado (idealmente, ya que solo Oracle, en sus aplicaciones orientadas a los negocios, tiende a optimizar la red).

Los conceptos de Virtualización y Aprovisionamiento aplican para los recursos computacionales de una organización: infraestructura, aplicación e información.

Infraestructura, en el sentido en que se crean redes virtuales de recursos computacionales (computadores, sistemas de almacenamiento, redes físicas, etc.) y se realiza el aprovisionamiento de las mismas cuando se ubica una tarea, mediante el software de administración de recursos, en tales recursos computacionales. En las aplicaciones, se presenta un fenómeno especial, estas pueden ir empaquetadas y desarrolladas en cualquier lenguaje de programación como servicios que son unidos mediante interfaces "Arquitectura SOA – Service Oriented Architecture"[7]. De tal manera se virtualizan los servicios ya sea por tipo o por funcionalidad según las necesidades del negocio y se aprovisiona de tal manera que se ubica el servicio que realiza la tarea especificada mediante el software administrador de recursos.

La Virtualización y el aprovisionamiento de la información se ve reflejada en que la información en un Grid debe llevar metadatos, es decir, el significado de la información, ya que la información puede estar en cualquier parte del Grid y solo es conectada mediante su significado o metadatos. Algunos autores comparan la información de un Grid como una Web Semántica; la W3C la define de la siguiente manera: "la Web semántica provee un marco común de trabajo que permite a los datos ser compartidos y reutilizados por las aplicaciones, las empresas y los limites de la comunidad"[8].

Algunos trabajos mencionan que hay varias cosas para contemplar en el término de Grid Computing. Tal vez uno de los aspectos más importantes en este nuevo paradigma es el de la seguridad de la información porque sin alguna infraestructura de seguridad (incluyendo herramientas de alto nivel) los usuarios no tomarán la ventaja computacional que ofrece Grid Computing. En este tema se puede decir que este paradigma, o más bien, arquitectura, no tendrá mucha acogida a menos que pueda ser segura, es decir, en términos de acceso, comunicación y algoritmos de encriptación de datos.

[7] Arquitectura orientada a servicios [en línea]: Definiciones SOA. Florida: Wikipedia Foundation, Inc., 2007. [Consultado 20 de diciembre de 2007]. Disponible en Internet: http://es.wikipedia.org/wiki/Arquitectura_orientada_a_servicios

[8] Servicios Web [en línea]. tecnologías inter-operativas. Asturias: W3C (World Wide Web Consortium), 2008. [Consultado el 20 de noviembre de 2007]. Disponible en Internet: http://www.w3c.es

Se puede mencionar el término "**Grid Secure Infraestructure** (GSI)"[9] como una aproximación a la solución esperada. Esta solución propuesta por el Global Grid Forum y la IETF como estándar de seguridad en Grid Computing está siendo implementada en varios sistemas de información tales como AFS, CVS, OpenSSH, MyProxy, etc.

Sin embargo, esto solo resuelve el problema de autenticación, o de saber si un usuario es quien dice ser. Muchos temas o retos y algunas herramientas de alto nivel todavía los han dejado de resolver. Por ejemplo, para que un usuario sea capaz de utilizar las herramientas de GSI[10] para acceder a las máquinas que hacen parte de dos diferentes bases de pruebas, se deben hacer algunos arreglos en los certificados de autenticidad para reconocer a cada usuario. Técnicamente esto es fácil de resolver, pero implica decisiones trascendentales en temas de seguridad.

Se puede destacar que las aplicaciones del Grid Computing son varias de acuerdo con la investigación e implementación que la organización desee realizar, es decir, la aplicación del Grid Computing va desde Computación Distribuida hasta entornos virtuales y procesamiento paralelo y en tiempo real, ya que esta arquitectura tiene varios aspectos que vale la pena mencionar, a saber:

- Confiabilidad,
- Eficacia,
- Efectividad,
- Escalabilidad,
- Capacidad de respuesta a fallos (mediante el aprovisionamiento y la correcta Virtualización).

Algunos estudios que se han realizado muestran que las definiciones realizadas anteriormente, así como las características y bondades que presenta Grid Computing[11] se pueden definir en las siguientes conclusiones:

- Coordinación de recursos que no están sujetos a un control centralizado.

[9] The Globus Toolkit [en línea]: Overview of the Grid Secure Infrastructure. Chicago: Globus, 2008. [Consultado 17 de enero de 2008]. Disponible en Internet: www.globus.org/security/overview.html
[10] GSI (Grid Secure Infraestructure) [en Linea]: Grid Security. Chicago: Globus Proyect, 2008. [Consultado 20 de enero de 2008]. Disponible en Internet: http://www.globus.org/security/overview.html
[11] What is the Grid. Op. cit., Disponible en internet: http://www.mellanox.com/pdf/whitepapers/GridTech_WhitePaper_final.pdf

- Utilización de estándares abiertos, protocolos e interfaces de propósito general.

Tomando como base lo anterior, se han encontrado varias herramientas que permiten la implementación de Grid Computing bajo diferentes enfoques o metas, ya que entre las que se encuentran en el mercado casi todas proporcionan los beneficios fundamentales de esta arquitectura. Algunas de estas herramientas son:

- Globus Toolkit, la cual es software libre y permite una modificación del código fuente para las necesidades específicas.

- Oracle 10g.

- Algunas aplicaciones de IBM, las cuales están diseñadas para propósitos especiales

o IBM® Grid and Grow™ Express (una solución para negocios competitivos). Ideal para principiantes en Grid Computing.

o Grid Medical Archive Solution (GMAS). Diseñado para el tratamiento de imágenes y almacenamiento de las mismas en el campo de la medicina específicamente.

o IBM® Grid Solution (Para procesamiento intensivo de datos). Ligado con WebSphere.

- Entre la aplicaciones más importantes de Sun Microsystems se pueden mencionar:

o Sun Grid Engine, conocida como N1 Grid Engine. Entre sus funciones se destacan el manejo de gran volumen de información de un Grid de más de mil computadores conectados en red.

o "CDO2, una aplicación comercial para la industria bancaria y firmas de investigación"[12].

- Algunas de las implementaciones más importantes en Grid Computing tienen que ver con Condor, sistema de software de alto desempeño que utiliza el poder computacional de toda una red para ejecutar trabajos de manera no centralizada.

[12] Sun Microsystems - SGE [en línea]: Sun Grid Engine. Santa Clara: SUN, 2007. [Consultado 29 de octubre de 2007]. Disponible en Internet: http://www.sun.com/software/gridware/

A continuación se muestra una breve descripción de los tres pioneros de la tecnología a nivel internacional:

Tabla 2. Principales investigadores en Grid Computing.

Ian Foster	• Director Del Instituto de Computación de la Universidad de Chicago. • Profesor de Ciencias de la computación en Argonne National Laboratory. • Lidera proyectos de ciencia de la computación desarrollando tecnologías de computación distribuida avanzada con el fin de resolver problemas en áreas de la ciencia que requieran alto nivel de computación. • Tiene múltiples premios por su investigación en computación distribuida.
Carl Kesselman	• Director del Centro para Tecnologías Grid en el Instituto de Ciencias de la Información. • Investigador en ciencias de la computación en la Universidad de California del Sur. • Su principal foco de investigación es Grid Computing ya que permite la creación de colaboraciones multiinstitucionales y organizaciones virtuales. • Junto a Ian Foster publicaron “Grid Services for Distributed System Integration” en el Año 2002.
Steve Tuecke	• Co-fundó la alianza de Globus, originalmente conocida como el proyecto de Globus, con Ian Foster y Carl Kesselman, era responsable de llevar la arquitectura, el diseño, y el desarrollo del software de Globus. • En 2002, Steve recibió el premio de TR100 de la revista Technology Review, que lo reconoció como uno de los máximos 100 innovadores jóvenes del mundo.

4. OBJETIVO GENERAL

Implementar una Malla Computacional (Grid Computing) experimental en uno de los laboratorios del Departamento de Ciencias de la Información de la Universidad Autónoma de Occidente.

5. OBJETIVOS ESPECIFICOS

- Seleccionar una herramienta para la implementación de la malla computacional.

- Diseñar y configurar una red de computadores que cumpla con los principios de funcionamiento de la computación en grilla en un laboratorio del Departamento de Ciencias de la Información.

- Implementar la malla computacional usando la herramienta seleccionada.

- Realizar la selección de un caso de estudio que permita realizar las pruebas de la implementación de Grid Computing en los laboratorios del Departamento de Ciencias de la Información.

- Desarrollar un plan de pruebas para la malla computacional implementada usando el caso de estudio determinado.

- Realizar una comparación de procesamiento de gran volumen de datos entre Grid Computing y una herramienta tradicional.

6. JUSTIFICACION

Se plantea este proyecto con el fin de implementar una herramienta que permita a la Universidad Autónoma de Occidente (UAO) aprovechar el nuevo paradigma denominado Computación en Malla o Grid Computing en el procesamiento de grandes volúmenes de datos. Los dos principios fundamentales del Grid Computing son virtualización (recursos individuales repartidos por tipo los cuales conforman un todo), y aprovisionamiento (como el sistema determina la necesidad del usuario mientras que se optimiza el rendimiento como un todo).

Mediante Grid Computing se mejorarán procesos como la simulación de procesamiento de información de gráficas de realidad virtual y la simulación de mecánica de fluidos, que son necesidades del departamento de Ciencias de la Información y de grupos de investigación que requieren un procesamiento de un gran volumen de datos, los cuales se están realizando actualmente en uno o varios servidores, pero desaprovechando el potencial de los demás computadores conectados en la red.

Grid Computing permite ser implementado inherente al sistema operativo de cada computador que pertenece a la malla (virtual), permitiendo que el software implementado pueda aprovechar los recursos ociosos de cada uno de estos computadores sin importar en que plataforma se esté trabajando.

Grid Computing representa una fase revolucionaria en la historia de arquitecturas distribuidas, algunos lectores la llamarían: una nueva tecnología perturbadora. Aunque es evidentemente una consecuencia de modelos de computación distribuida ya previamente establecidos.

La implementación de una malla computacional le permitirá a los grupos de investigación y a los usuarios en general las siguientes características:

- Nuevos estándares de comunicación para lenguajes orientados a objetos que hacen más fácil la construcción de sistemas de información para redes de arquitectura distribuida.

- Los microprocesadores de alto rendimiento estarán disponibles, haciendo posible desarrollar e implementar aplicaciones a gran escala en un número considerable de computadores de bajo costo en vez de un computador de alto costo.

- Tecnología de red de alta velocidad se hace posible a un bajo costo y fácilmente disponible, ofreciendo un alto rendimiento cuando se desarrollen aplicaciones de arquitectura distribuida.

Además, existe la posibilidad de empezar a crear cultura de implementación de nuevas tecnologías que en el mundo están tomando fuerza ya que esta nueva arquitectura permite, como se menciona anteriormente, aumentar la capacidad de cómputo de las compañías al mismo tiempo que se disminuyen los costos de las inversiones en hardware.

Un aporte muy importante de la implementación del Grid Computing en la Universidad Autónoma de Occidente es entender la arquitectura y el funcionamiento de este nuevo paradigma, con el fin de conocer que tanto los estudiantes de la universidad, profesores y usuarios en general, puedan aprovechar todos los recursos ociosos de la amplia red que ofrece la universidad y así hacer que las investigaciones y los trabajos que impliquen un alto grado de procesos de computación sean más eficientes y más efectivas a la hora de llevarlas a cabo.

Finalmente, grandes empresas multinacionales como IBM, "Microsoft"[13], IDC, "Oracle"[14], entre otras, ya han empezado a implementar todos los procesos mediante el Grid Computing debido a las grandes bondades que esta ofrece, lo que nos conlleva a pensar que las personas o estudiantes en la Universidad que manejen herramientas de Grid Computing tendrán un margen de competitividad mayor en el ámbito laboral y una mejor comprensión de los procesos de negocio que una compañía puede tener.

[13] Microsoft [en línea]: For IT Professionals. Barcelona: Microsoft Inc., 2008. [Consultado 2 de octubre de 2007]. Disponible en Internet: http://www.microsoft.com

[14] Oracle 10g [en linea]: Oracle Grid Computing. New York: Oracle, 2008 [Consultado 15 de febrero de 2008]. Disponible en Internet: Http://www.oracle.com/grid

7. METODOLOGIA

Para lograr los objetivos planteados en este proyecto se propuso:

- Documentación del estado del arte en Grid Computing. La meta de este capítulo es la completa documentación de los fundamentos de Grid Computing, conocer el estado del arte, entre los cuales podemos mencionar la virtualización y el aprovisionamiento de los recursos computacionales.

- Selección de herramienta para la implementación de la malla computacional. Se tratarán temas como las consideraciones que se deben tomar en cuenta en el diseño de una malla computacional, tales como la seguridad, el diseño, la confiabilidad, entre otros.

- Recolección de información del laboratorio. En la recolección de información se pretende observar la infraestructura tecnología (ambiente) en la cual se va a desarrollar el proyecto, es decir, que plataformas de software y de hardware existen actualmente en los laboratorios del departamento de Ciencias de la Información, ya que este es un punto de vital importancia en la escogencia de la herramienta que permita implementar Grid Computing en la Universidad. También se debe tener en cuenta los convenios que existen con los diferentes proveedores de software, ya que hay herramientas tanto de software libre como de versiones comerciales, tal es el caso de Oracle 10g, y herramientas de IBM,

- Implementación de la herramienta. Implementación de la herramienta que nos permita resolver el caso de estudio de una manera eficaz y eficiente con el fin de que se cumplan todos los objetivos.

- Selección del Caso de Estudio. Uno de los principales objetivos es determinar el caso de estudio, ya que teniendo como base los problemas que se van a solucionar, los beneficios que se van a obtener y bajo que plataforma se va a implementar, se puede llevar a cabo la totalidad del proyecto y en el mejor de los casos seguir con la investigación.

- Comparación entre computación normal, y Grid Computing. Realizar la comparación de procesos de alta carga computacional en las dos arquitecturas, para observar el progreso y la meta final que es el incremento del rendimiento en procesamiento de información en un ambiente de alta carga computacional.

- Análisis de los resultados. Una vez terminado el proyecto se analizó los principales resultados y las conclusiones en cuanto a su realización, revisión, etc.

8 DESARROLLO DEL PROYECTO

El desarrollo del proyecto va ligado con las etapas que se mencionan en la metodología escogida para la realización del proyecto y se enumeran de la siguiente manera:

8.1 DOCUMENTACION DEL ESTADO DEL ARTE EN GRID COMPUTING.

El estado del arte muestra el estado actual de la tecnología a implementar en la Universidad. Como se ve con anterioridad Grid Computing es una tecnología realmente nueva que muestra a la comunidad científica un avance significativo en poder computacional para investigaciones que así lo requieran.

Aquí se presentan un conjunto de clusters que interrelacionan entre si para formar un solo Grid. Cada Cluster puede ser visto de diferentes maneras. Puede ser visto como una organización virtual o como un conjunto de recursos computacionales que proveen sus recursos para que tareas que se soliciten puedan ser ejecutadas.

El concepto de Grid Computing, encierra mucho mas allá de comunicación entre diferentes dominios, y diferentes tipos de arquitecturas, el termino Grid tiene muy en cuenta el concepto de heterogeneidad de recursos tanto de hardware como de software.

En el caso de este trabajo de grado, se presentara dos computadores (un maestro y un esclavo) con diferentes características de hardware con el fin de alcanzar una semejanza con un Grid, es decir, se implementa la base de un Grid, un Cluster.

- Investigaciones en Grid Computing en Colombia. Las mayores investigaciones se pueden enumerar en dos aspectos: Grid Colombia mediante la red RENATA, y el proyecto CampusGrid en la Universidad De Los Andes. Grid Colombia es una organización virtual en proceso de formación, con una base fundamentalmente académica, destinada originalmente a centralizar los esfuerzos para la creación de una malla computacional de propósito académico en Colombia usando las Redes de Tecnología Avanzada (RENATA) a nivel regional. Algunas de las actividades que han desarrollado son las de conferencias a nivel mundial con el fin de compartir experiencias, dar a conocer los avances científicos, y promover la investigación en el tema; otras de las actividades que promueve Grid Colombia son los seminarios de implementaciones de Clusters y puesta en marcha de sistemas operativos en donde operan las herramientas que soportan esta tecnología.

Por otro lado, en Colombia las mayores investigaciones se realizan en la Universidad de los Andes, donde recientemente uno de los principales investigadores de la Universidad de Chicago (Borja Sotomayor) dicto una conferencia sobre Grid Computing y sobre los beneficios que esta ofrece.

Lo que se pudo constatar fue que actualmente en Colombia no se ha creado un Grid como tal, ya que este concepto va mas allá de colocar a funcionar los recursos computacionales de varios computadores colocados en red. El concepto de Grid está ligado a la heterogeneidad de los recursos, y a la administración no centralizada, este último quiere decir que cada recurso o cada cluster es dueño de si mismo y no pertenecen a un mismo dominio.

Además, la definición de Grid que se pudo deducir, según la explicación realizada por el profesor Harold Castro de la Universidad de los Andes, a grandes rasgos es la puesta en marcha de recursos computacionales, infrautilizados/subutilizados o no, en forma de clusters donde cada uno de éstos están en diferente dominio y poseen características diferentes. Esta definición fue lograda a partir de las investigaciones que se han realizado en esta universidad.

Las principales investigaciones realizadas en la Universidad de los Andes están basadas en mejorar los tiempos de computación de algoritmos que requieren alta carga computacional, como soluciones al problema de QUADRATIC SNAPSACK e implementaciones de software que administre de la mejor manera posible el Cluster que se implementó en dicha Universidad.

8.2 SELECCIÓN DE LA HERRAMIENTA.

La herramienta seleccionada fue Condor versión 6.8.6.

Para la selección de la herramienta que permitió la implementación del Grid a un nivel de Cluster se tuvo en cuenta varios factores, los cuales se mencionan a continuación:

- Documentación y soporte. Debido a que la tecnología Grid es un tema que no lleva mucho tiempo en el campo de la investigación, y la cantidad de personas relacionadas en el tema y la documentación encontrada acerca del software de administración del Grid a implementar es relativamente escasa, además de poco confiable (la mayoría de los manuales estas desarrollados con base del ensayo y error); este fue un punto importante a la hora de escoger el software a implementar en los laboratorios.

Los proyectos en Colombia empiezan desde la base de su investigación, en este caso es el funcionamiento de un cluster y de la "computación de alto desempeño"[15].

Según las investigaciones realizadas en el levantamiento de información del proyecto, los principales investigadores en el área de Grid Computing en Colombia, ubicadas en Bogota, son las personas encargadas de ofrecer la información relacionada con este tema, ya que la idea principal de ellos, es la implementación de Grid a nivel nacional mediante conferencias y Workshops los cuales permiten intercambiar ideas y/ó avances logrados mediante la puesta a punto de esta nueva arquitectura. Los estudios preliminares de estos investigadores los llevo a tener como meta hacer parte del proyecto EELA donde el software de "administración de los clusters es Condor"[16].

El soporte encontrado en la recolección de información de las herramientas relacionadas fue más que todo relacionado en foros, los cuales son conformados por personas que son estudiantes y profesionales, los cuales tienen proyectos de iniciación a la computación en Grid.

- Futura investigación. Las investigaciones que se pueden realizar en el campo de la computación en Grid van de acuerdo con la filosofía que adopte una organización, bien sea comercial, universitaria, desarrollo, investigativa, seguridad, etc.; ya que cada una puede usar la computación en Grid en Pro de sus necesidades, Ej.: en el ámbito comercial, cuya aplicación establece el comportamiento de la administración de la cadena de abastecimiento en la industria, requieren un software que gestione una gran cantidad de información no solo de cada una de las organizaciones sino de las organizaciones como un conjunto, ya que según los lineamientos de SCM (Supply Chain Management), las organizaciones comparten información para optimizar sus procesos. Esta cantidad de información en el sector comercial es realmente grande y requiere de recursos computacionales que en algunos casos no son fáciles de conseguir, ya sea por presupuesto o porque la plataforma que se maneje en la empresa no soporte esa tecnología. En el ámbito investigativo, se pueden mencionar temas como seguridad informática, desarrollo de algoritmos de orden exponencial, entre otros los cuales, Cóndor, ofrece la capacidad de utilizar al máximo los computadores que hagan parte del Grid para ejecutar dichas tareas de manera simultánea o de manera paralela.

Se pueden mencionar en algunos lineamientos científicos, los cuales se enumeran de la siguiente manera:

[15] HTC,High [en línea]: Throughput Computing and its requirements. Wisconsin: Universidad de Wisconsin, 2008. [Consultado 16 de enero de 2008]. Disponible en Internet: http://www.cs.wisc.edu/condor/manual/v6.8/1_1High_Throughput_Computin.html#804

[16] Manual de Condor [en linea]: Manual de usuario. Wisconsin: University of Wisconsin-Madison, 2008. [Consultado 19 de diciembre de 2007]. Disponible en Internet: http://www.cs.wisc.edu/condor/manual/v6.8/ref.html

- Descubrir vida en otros planetas.

- Optimización de recursos computacionales.

- Desarrollo de nuevas aplicaciones de ejecución en paralelo

- Mejoramiento de tiempos de ejecución de tareas de computación de alto desempeño.

- Análisis de información de desastres naturales como tifones, terremotos, erupciones volcánicas.

- Software libre o de Código Abierto. Aunque existen soluciones comerciales que incorporan técnicas de agrupación de máquinas, aquellas que gozan de una mayor aceptación en la comunidad científica siguen siendo las que proponen la utilización de software libre o abierto. La construcción de clusters permite que la implementación de software libre mejore la relación calidad/precio.

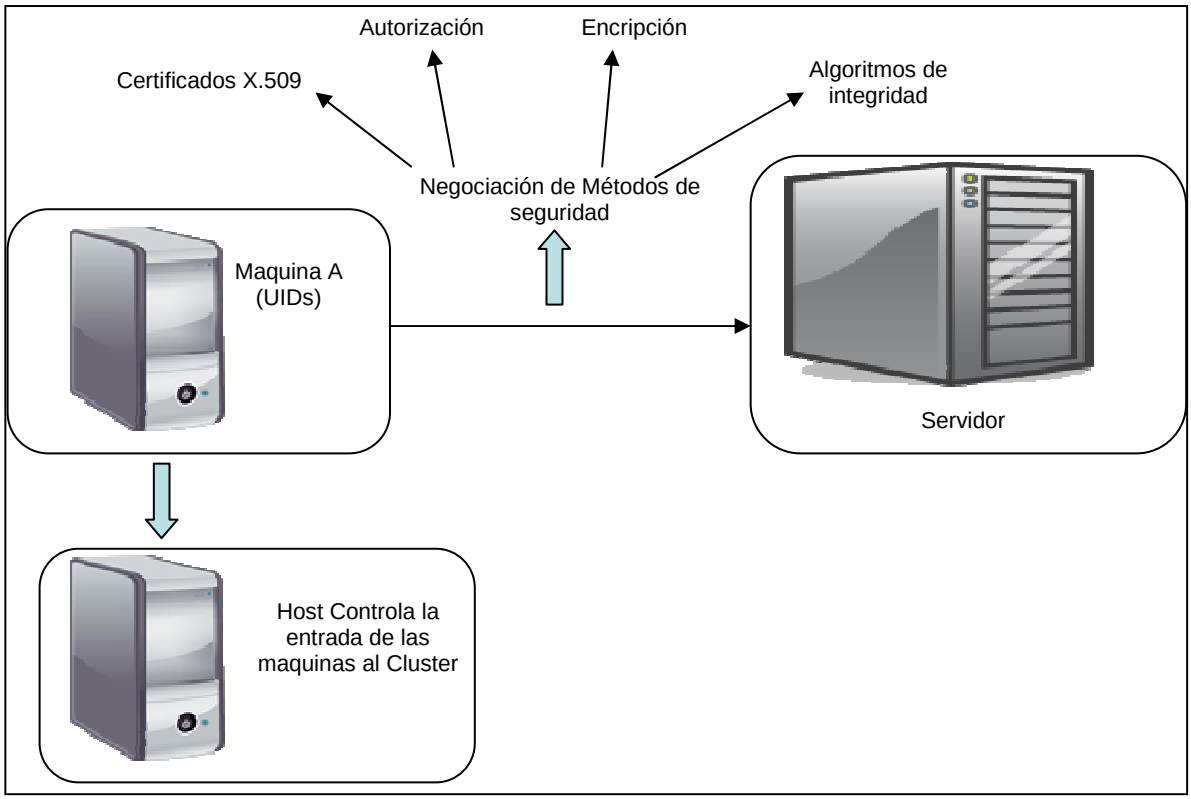
- Seguridad. Condor se encarga de mantener la seguridad en el momento en que un usuario va a autenticarse en el sistema, ya que Condor evita que usuarios no autorizados se autentiquen en el sistema, además provee métodos de seguridad, los cuales son:

- Cuentas de usuario en Condor. Condor usa los UID (User IDentification numbers) los cuales permite monitorear los usuarios que están ejecutando una o unas tareas especificas.

- Configuración básica. Un computador es el encargado de tener la registradas las maquinas que tienen acceso a la información de Condor (mediante dominios). Es limitante ya que trabaja con direcciones IP y por tanto lo que se configura son las maquinas.

En la siguiente figura se puede observar los componentes que intervienen en la seguridad de Condor:

Figura 1. Componentes que intervienen en la seguridad de Condor



Fuente: PARDO, María Carolina. Evaluación de las diferentes herramientas utilizadas en Grid, para la implementación del proyecto EVA-R GRID. Bogota, 2006. p. 79. Trabajo de Grado (Ingeniero de Sistemas). Universidad del Bosque. Facultad de Ingeniería.

Es una realidad que el software libre está cambiando la forma de concebir las cosas, está posibilitando el desarrollo de sistemas de alto rendimiento, fiabilidad y productividad a un coste muy bajo. El software libre ayuda a la creación de un entorno de computación en Grid a muy bajo costo, así como de apuntar las líneas de desarrollo que marcarán el futuro del cómputo de grandes números.

Mediante los puntos anteriores se escogió la herramienta Condor porque, de manera general, es la que más se adecua a las necesidades de este proyecto, ya que con esta herramienta no solo se implementa una malla computacional experimental en la Universidad Autónoma de Occidente sino que se realiza un acercamiento a la filosofía que se maneja en la implementación del Grid.

En el siguiente cuadro se mostrara un resumen de las principales características del software escogido para la implementación en el laboratorio.

Tabla 3. Características principales de Condor.

ASPECTO	DESCRIPCION
Seguridad	• Manejo de UID • 4 tipos de autenticación (Certificados, Autorización, Encripción, Algoritmos de integridad)
Administración de Recursos	• La maquina central es la encargada de negociar y hacer el match según los recursos disponibles. • Los recursos son manejados mediante classAds que actúan como clasificados de un periódico, lo que permite que la maquina central identifique de una manera optima los recursos disponibles. • Mediante el condor_collector se recogen las classAds enviadas por los demonios de las maquinas esclavas.
Monitorización de tareas	• Condor_schedd permite observar las tareas que se estan ejecutando en el sistema.
Funcionalidad en el sistema	• Condor_master • Condor_startd • Condor_starter • Condor_schedd • Condor_shadow • Condor_collector • Condor_negotiator.
Arquitectura del Sistema	• Administrador Central (Master de Condor) • Ejecucion • Registro
Unión diferentes Clusters de Condor	Condor permite que tareas que se solicitaron en el Cluster A se ejecuten en el Cluster B de Condor, ya que pueda que una tarea no encuentre recursos disponibles para ser ejecutadas.

8.3 RECOLECCION DE INFORMACIÓN DEL LABORATORIO.

Mediante la recolección de información de los recursos computacionales se pudo hacer un esquema de cuales eran los recursos computacionales que se necesitaban para la implementación del Grid, así como la herramienta a implementar, en la Universidad.

Debido a que este proyecto es proceso de implementación de una malla computacional experimental, se recomendó trabajar con más de 2 computadores de diferentes características, pero solo se trabajo con 2 computadores, pero finalizando el proyecto se logro incluir en el proyecto otro computador.

A continuación, se muestra la información recolectada del laboratorio de informática en el que se estableció un ambiente de trabajo lo más cercano posible a un Grid.

Software encontrado en el laboratorio.

- Linux Fedora Core 5.
- Apache.
- Tomcat.
- Eclipse.
- JDK 1.4.
- Jcreator.
- Oracle 9i.
- Windows XP SP2.

Hardware encontrado en el laboratorio.

Computadores IBM – 6349 – 94S.

- Disco Duro de 120 G.
- Memoria RAM de 1GB DDR2.
- Procesador de 1.6 GHz.
- Tarjeta de video NVIDIA 16 MB.

- Tarjeta de red Intel 845i.

Finalizando la implementación de la malla computacional experimental en los laboratorios de la Universidad Autónoma de Occidente, se logró implementar en otro computador cuyas características de hardware se describen a continuación.

Computador Dell Optiplex GX520.

- Disco Duro de 80 GB.

- Memoria RAM de 1 GB DDR2.

- Procesador de 3GHz Dual Core.

- Tarjeta de video Intel 82945 G.

- Tarjeta de red BroadCom NetExtreme 57X Gigabit.

La recolección del hardware encontrado permitió establecer un ambiente de trabajo propicio para el desarrollo del proyecto, éste fue organizado de la siguiente manera:

Los tres computadores tienen el sistema operativo Fedora Core 5, con el software de administración del Grid (Cóndor).

8.4 MODELO DE CÓMPUTO.

El modelo de cómputo que se implementó en los laboratorios de la Universidad Autónoma de Occidente se realizo de acuerdo a la recolección de información y a la selección de la herramienta que se realizó.

Para diseñar el modelo de cómputo se analizo de qué manera operaba el software que fue implementado.

En esta herramienta se debe especificar muy bien quien va a ser el computador esclavo ya que este es único para el cluster que se va a implementar, tal como se indica en la figura 2.

Figura 2. Arquitectura de Condor

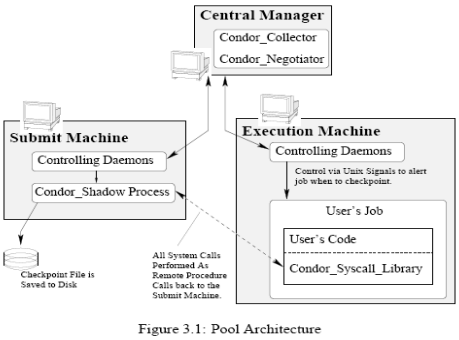
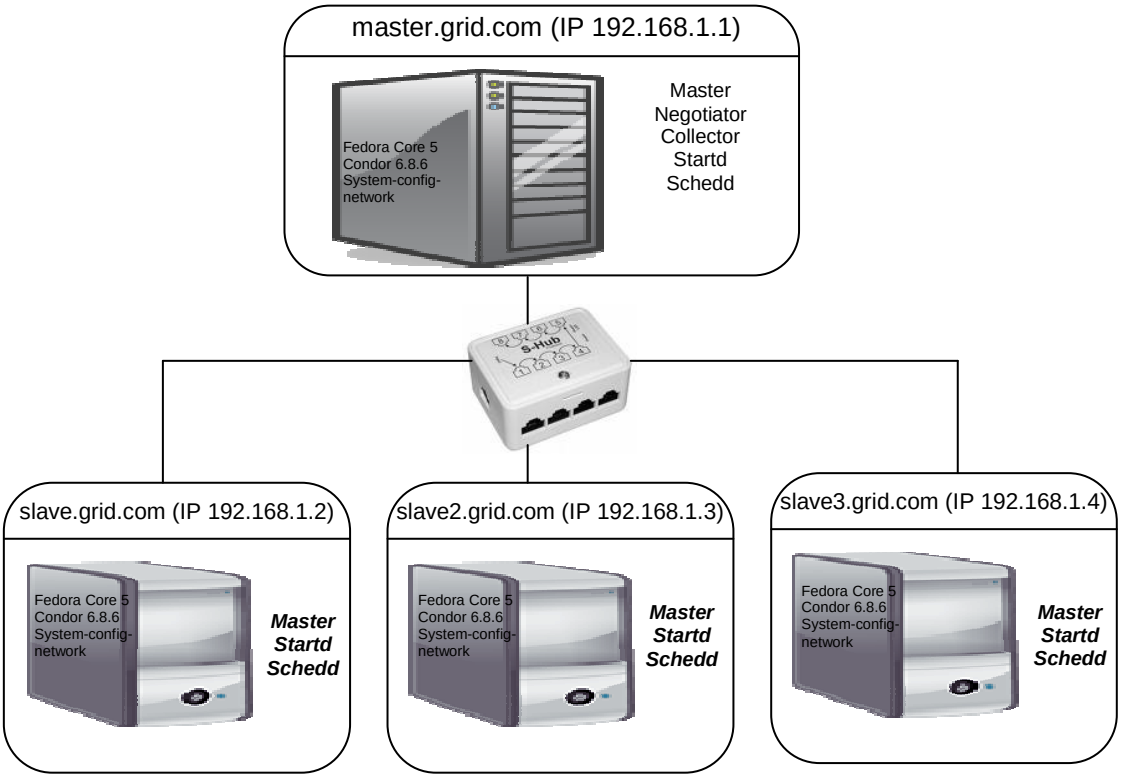


Figure 3.1: Pool Architecture

Fuente: Manual de Condor versión 6.8.4 [en linea]: Arquitectura de Condor. Sydney: Mathematics Department. Macquarie University, 2000 [Consultado 18 de septiembre de 2007]. Disponible en Internet: http://www.cs.wisc.edu/condor/manual/v6.8/3_1Introduction.html

Según la arquitectura que debe seguir Condor, y algunas consideraciones que se describen en la siguiente sección, y los recursos disponibles para el proyecto, se concluyó que la mejor arquitectura de hardware para la implementación de la herramienta y las posteriores pruebas que se hacen sobre la misma esta dada por la siguiente figura.

Figura 3. Arquitectura diseñada para la implementación de Condor



En esta figura se puede observar que esta compuesta por un maestro que puede tanto asignar tareas como ejecutarlas.

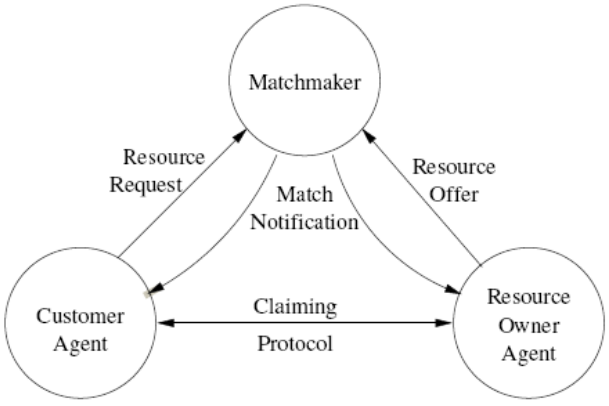
El maestro debe ser quien escoge que o cuales de sus esclavos realizan la tarea que uno o varios de sus esclavos quieren realizar. En el maestro debe ir el maestro del software de Grid configurado de tal manera que reconozca a cada uno de los esclavos y la capacidad computacional que cada uno de estos tiene.

El esclavo debe estar configurado de la siguiente manera: debe tener la aplicación que se va a ejecutar en el Grid, para este caso debe tener instalado Condor, y debe estar configurado el software de Grid configurado para reconocer su maestro. La aplicación que se va a correr en el Grid idealmente debería ser divisible, debido a que esta característica permite ejecutar la aplicación al mismo tiempo en varios computadores del Grid con diferentes parámetros lo que ocasionaría una disminución en el tiempo de respuesta en la ejecución de un algoritmo.

Con respecto al software se implemento la herramienta de acuerdo a la arquitectura diseñada, ya que la implementación del software va muy ligada a la arquitectura que se implantó en el laboratorio.

La arquitectura de administración de recursos de condor como esta especificada en la figura 4, permite visualizar el funcionamiento del proceso que hace una tarea en un computador específico.

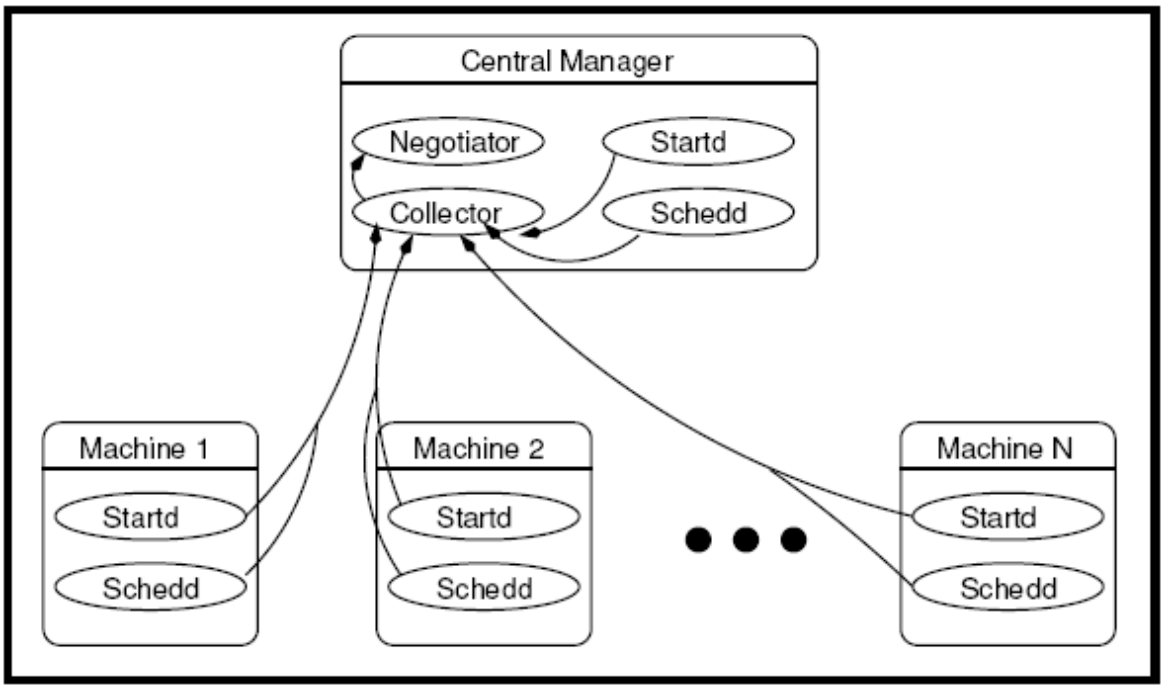
Figura 4: Arquitectura de administración de recursos de Condor



Fuente: Deploying a Hight Throughput Computing Cluster [en linea]: Arquitectura de administracion de recursos en Condor. Wisconsin: Departamento de Ciencias de la Computacion Universidad de Wisconsin, 2000. [Consultado 15 de enero de 2008]. Disponible en Línea: http://www.cs.wisc.edu/condor/doc/hpcc-chapter.pdf

En la siguiente figura se muestra cual es el recorrido que una tarea sigue en todo el cluster que se implementó, este recorrido puede empezar en una maquina submit la cual es la encargada de hacer la solicitud, esta viaja hasta el maestro y este decide que computador o computadores pueden ejecutar dicha tarea, estas son las razones por las cuales el master de Condor no debe ejecutar tareas.

Figura 5. Descripción de los demonios en Condor.



Fuente: Deploying a High Throughput Computing Cluster [en linea]: Distributed Job Scheduler. Wisconsin: Universidad de Wisconsin. Departamento de Ciencias de la Computacion, 2000. [Consultado 10 de Diciembre de 2007]. Disponible en línea: http://www.cs.wisc.edu/condor/doc/beowulf-chapter-rev1.pdf

En el recorrido que se muestra en la figura 5 se observan los demonios principales que interactúan en el sistema Condor, estos demonios son:

- Master. Es el responsable de que los demonios se estén ejecutando en cada maquina. Monitorea los demás demonios y chequea periódicamente si cada uno de ellos presenta algún tipo de fallas, si es así, se genera un punto de chequeo y re reinicia el demonio afectado. Permite reconfigurar los demonios remotamente.

- Startd. Representa los recursos dentro del sistema Condor y muestra los atributos que tiene. Cuando este demonio se inicia, el sistema Condor esta preparado para ejecutar

cualquier tarea; este demonio se puede ejecutar en cualquier maquina del sistema Condor.

- Schedd. Este demonio es el que almacena y administra la tarea ya registrada, luego de verificar el estado de los recursos existentes en el grupo Condor. Cada computador que este encargado de ejecutar tareas debe tener este demonio iniciado ya que es el encargado de anunciar las tareas pendientes en cola y es el responsable de solicitar recursos disponibles para satisfacer las solicitudes.

- Collector. Es el encargado de recoger toda la información del sistema Condor, a este demonio llegan todas las classAds enviados por todos los demonios, con excepción del Negotiator.

- Negotiator. Realiza el match de las tareas registradas en el grupo y los recursos disponibles en el sistema, este demonio solicita al Collector el estado actual del sistema y según los datos enviados el Negotiator hace el empate entre la solicitud y el recurso. Este demonio contacta al Schedd de la maquina donde se ejecutara la tarea, este la registra y la administra según el estado de su maquina.

8.5 IMPLEMENTACION DE LA HERRAMIENTA (CONDOR V6.8.6).

Antes de la implementación de Cóndor se realizaron una serie de análisis que permitiera una implementación adecuada y acorde a las necesidades del presente proyecto. Las consideraciones tenidas en cuenta fueron las siguientes:

- ¿Bajo qué sistema operativo implementar Cóndor?. “El sistema operativo sugerido en la investigación realizada en la Universidad de los Andes, en el departamento de sistemas de información fue la distribución Scientific Linux, pero se decidió trabajar en Fedora Core 5, porque una de las sugerencias fue que la implementación de Cóndor tiene que estar precedida de conocimientos del sistema operativo donde este va a ser implementado. Además que para el ámbito investigativo, el Fedora Core 5 permite una fácil configuración de las variables de entorno, del sistema de archivo, de montaje de volúmenes, y el montaje de aplicativos que permitan una fácil interacción con Cóndor”[17].

- ¿Direcciones IP fijas o dinámicas (servidor DNS)?. Ya que el Cluster es pequeño, y que Cóndor en su configuración hay variables como la UID_DOMAIN Y LA GID_DOMAIN que exigen un dominio tanto del usuario como el grupo que son propietarios de Cóndor, se trabajo con DNS, donde el DNS primario va a ser el mismo

[17] Manual de Condor [en línea]: Caracteristicas de Condor. Wisconsin: Universidad de Wisconsin, 2008. [Consultado el 28 de septiembre de 2007]. Disponible en Internet : http://www.cs.wisc.edu/condor/manual/v6.8/ref.html

Master de Cóndor, pero que no necesariamente tiene que ser el computador con mejores características de hardware.

- ¿Qué características debe cumplir la máquina central? . En cuanto a hardware, la máquina central (CONDOR_HOST) no debe tener mucha potencia, de hecho, se decidió que el mejor computador actuara como un esclavo, para probar que la configuración de Cóndor quedara de tal manera que asigne primero al computador de mejores características. "Las características principales que debe cumplir la máquina central de Cóndor es que se deben correr en esa máquina, los demonios suficientes para la negociación de recursos que hacen parte del cluster"[18]. En esta máquina central quedará el repositorio de información del cluster, donde también se hará la asignación de recursos disponibles con los trabajos que han sido encolados desde un esclavo o desde el mismo maestro.

Una consideración importante es la que se especifica en el manual del administrador: "Un Grid con 100 maquinas requerirá aproximadamente 25 Mbytes de memoria para las tareas del master. Un Grid con 1000 maquinas requerirá aproximadamente de 100 Mbytes de memoria para las tareas del master"[19].

- Desde qué máquinas se ejecutarán (encolar) trabajos en el Grid ?. El sistema Cóndor provee un mecanismo que permite distinguir a las máquinas que puedan encolar un trabajo, este mecanismo permite diferenciar dichas máquinas por medio de direcciones IP o por medio de dominios. Esta configuración se realiza por el archivo de configuración de ambiente por medio de la variable HOSTALLOW_WRITE y HOSTALLOW_READ. Si el Sistema Cóndor está protegido por un firewall, estas variables pueden ir con *, ya que es el firewall quien decide que máquina puede leer o escribir sobre los archivos de Cóndor. Es muy importante establecer las máquinas que pueden acceder al sistema, ya que si éste se comunica con otros cluster a través de Internet, usuarios diferentes a los permitidos pueden aprovechar la capacidad computacional y ejecutar procesos sin los permisos requeridos.

Para efectos del presente proyecto, se procedió a colocar las variables de forma tal que acepten todos los computadores que hacen parte de la red que se implementó, ya que esta sólo está conformada por los 3 computadores disponibles y no está conectada a ninguna red externa.

Ejecutar Condor como súper usuario (root) o no?. "Se recomienda que los demonios que se corran en Cóndor se corran como root en las maquinas que sean maestras, en este caso la maquina maestra. Y los demás maquinas como los esclavos se deben correr como los usuarios de cóndor diferentes de root"[20].

[18] Ibíd., Disponible en Internet : http://www.cs.wisc.edu/condor/manual/v6.8/ref.html
[19] Ibíd., Disponible en Internet : http://www.cs.wisc.edu/condor/manual/v6.8/ref.html
[20] Ibíd., Disponible en Internet : http://www.cs.wisc.edu/condor/manual/v6.8/ref.html

- En qué carpetas debe ir la configuración de Condor según el sistema de archivos del sistema operativo? . Las carpetas de Cóndor deben ser únicas en cada máquina que haga parte del Cluster o Grid. Dichas carpetas principales son Log, Spool y Execute. Estas carpetas contienen archivos de log en los cuales se registran todos los procesos que ejecutan los demonios que hacen parte de cada máquina. También contienen información acerca de la configuración de cada máquina, así como la carpeta execute, la cual funciona como la carpeta de trabajo actual para todos los trabajos de Cóndor que trabajen en una máquina en particular.

8.5.1 Configuración de Condor. Para la instalación de Cóndor se siguió la guía de instalación del "Condor Project"[21], y a continuación se presentan las características particulares del cluster construido con Cóndor.

Para administrar recursos computacionales a través de Condor, es necesario habilitar 3 clases de tareas:

- Administración.

- Envío de Tareas.

- De ejecución de Tareas.

El cluster fue diseñado para tener un nodo responsable únicamente de las tareas de administración y un nodo proveedor de recursos, que ofrece al tiempo las tareas de envío y ejecución de tareas. El master de Condor fue instalado en el Nodo Maestro, mientras las funciones de envío y ejecución fueron habilitadas en el único Nodo Esclavo. Este diseño del cluster requirió la siguiente sentencia de instalación.

- Para el maestro.

```
condor_configure --install=/opt/condor-6.8.6/release.tar --
install-dir=/opt/condor-6.8.6 --local-dir=/opt/condor-6.8.6 -
-type=central-manager,submit --owner=master, root
```
[22]

Las 3 primeras opciones indican sencillamente los directorios de instalación del scheduler o planificador. El valor del atributo type indica que el nodo se comportará como

[21] Ibíd., Disponible en Internet : http://www.cs.wisc.edu/condor/manual/v6.8/ref.html
[22] Ibíd., Disponible en Internet : http://www.cs.wisc.edu/condor/manual/v6.8/ref.html

administrador del cluster y el atributo owner se refiere a que el usuario predeterminado de Condor es el usuario master que fue creado para este propósito.

A pesar de que la instalación es realizada usando la cuenta del usuario root por asuntos de seguridad, Condor internamente consigue que los demonios de la aplicación sean iniciados y ejecutados por el usuario especial master (creado en la instalación de Condor), el usuario root no tiene permitido, por ejemplo, enviar trabajos.

- Para el esclavo.

```
condor_configure --install=/opt/condor-6.8.6 --install-
dir=/opt/condor-6.8.6 --local-dir=/opt/condor-6.8.6 --
type=execute, submit --owner=slave, root
```

A diferencia de la instalación del maestro, el atributo type es execute, submit para habilitar las tareas de envío y ejecución de trabajos en la maquina esclavo. Esto quiere decir que esta máquina puede tanto solicitar recursos al cluster cómo proveerlos.

Luego de instalado Condor, tanto en el nodo Maestro como en el Esclavo, se deben configurar modificando algunos parámetros el los archivos de configuración local y global de cada maquina. Las siguientes son las modificaciones incluidas en tales archivos:

La configuración del Maestro esta caracterizada por los siguientes ítems:

- En el maestro y solo en el maestro se corren los demonios de condor_collector y condor_negotiator.

- La variable de entorno CÓNDOR_HOST debe llevar el nombre del maestro, es decir, de él mismo.

- La variable condor_config se debe adicionar en el archivo /etc/.bashrc

La configuración del nodo esclavo implementado en Cóndor se caracteriza por:

- En los esclavos solo se ejecutarán los demonios startd, master, schedd.

- Se configuraron como máquinas tanto de submit como de execute.

- La variable de entorno CÓNDOR_HOST especificada en el archivo de configuración local se le direccionó (apunto) hacia el maestro.

- La variable condor_config se debe adicionar en el archivo de configuración del sistema /etc/.bashrc

8.5.2 Errores y/o preguntas con su solución encontrada durante la implementación. Aquí se presentan algunas soluciones encontradas (ensayo y error, o por ayuda de investigadores de la Universidad de los Andes) durante la implementación.

Tabla 4. Posibles errores en la configuración del cluster de Condor.

ERROR	SOLUCION
Se puede utilizar los aplicativos bindconf o system-config-network para la configuración de las direcciones IP de los computadores?	es simplemente editar los archivos /etc/host tanto el maestro como el esclavo debe tener las dos líneas: <ipmaestro>maestro.uautonoma.edu.co maestro <ipesclavo>esclavo.uautonoma.edu.co esclavo y /etc/network/interfaces gmaster:~# cat /etc/network/interfaces # This file describes the network interfaces available on your system # and how to activate them. # The loopback network interface auto lo iface lo inet loopback # The primary network interface auto eth0 iface eth0 inet static address <ip de la maquina> netmask 255.255.255.0 #porque tienes sólo dos maquina. gateway <puerta de salida la maquina>
Dónde se coloca la variable de entorno	Dentro de /etc /bashrc

CONDOR_CONFIG=/etc/condor/etc/condor_config ?	
Se tienen el siguiente archivo > el Hello.cmd es asi: > > ##################### # # Example 1 # > Execute a single Java class, # not on a shared file > system # #################### > universe = java > executable = Hello.class > arguments = Hello > output = Hello.output > error = Hello.error > should_transfer_files = YES > when_to_transfer_output = ON_EXIT > queue (AQUI ES EL PROBLEMA) > #./condor_submit > /path/Hello.cmd submitting job:failed to transfer executable > file Hello.cmd?	La respuesta de este problema es un conjunto de procesos que se deben hacer: primero, deshabilitar el SELinux, después se debe verificar que el usuario que envía el trabajo debe existir en la maquina de ejecución y tener la misma ruta del trabajo.
Por qué sale el error (cuando se realiza un submit a algún trabajo): > Failed to transfer the executable file Hello.submit.	Este error regularmente se presenta cuando el archivo ejecutable no existe, el ejecutable y el referenciado en el .submit, pero no es el mismo No existe un usuario como el que envía el trabajo en el PC de execute.
Cuando se ejecuta en el esclavo, el comando ./condor_status sale el siguiente error: que no reconoce el Collector en el maestro	Con absoluta certeza no hay comunicación bidireccional entre maestro y esclavo. Se debe revisar la configuración de etc/hosts.

8.6 SELECCIÓN DEL CASO DE ESTUDIO - LA VUELTA DEL CABALLO.

8.6.1 Introducción. El siguiente constituye un caso de estudio sencillo cuya finalidad es observar el mejoramiento de los tiempos de respuesta de los algoritmos que se ejecutan en un ambiente de Grid, versus los algoritmos que se ejecutan en un ambiente tradicional.

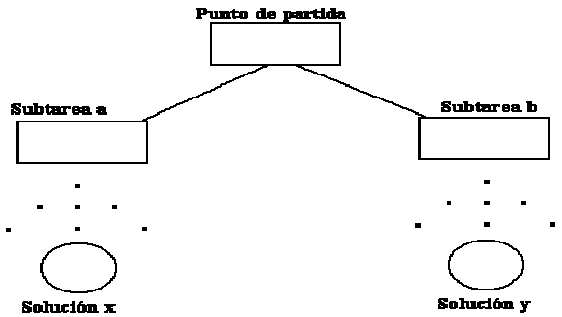
El algoritmo utilizado en este caso de estudio representa un ejemplo de un algoritmo de vuelta atrás (backtracking) propuesto por "Wirth"[23], el cual consta de un tablero de NxN posiciones (donde N es el numero de casillas hacia la izquierda y hacia la derecha) y una ficha que se mueve en dicho tablero con los mismos movimientos del caballo en el juego de ajedrez.

Se analizaran los tiempos de ejecución de este algoritmo mas no se explicara el algoritmo como tal, ya que el objetivo principal de este caso es demostrar que la computación en Grid mejora la ejecución de los algoritmos que exigen una alta carga computacional.

8.6.2 Algoritmos de Backtracking (vuelta atrás). Los algoritmos de vuelta atrás se utilizan para encontrar soluciones a problemas cuya solución solo se llega explorando todos los caminos posibles y que para explorar dichos caminos hay que devolverse a una solución anterior e intentar llegar a otra solución con diferentes parámetros u movimientos. No siguen unas reglas para la búsqueda de la solución, simplemente una búsqueda sistemática, lo que significa que hay que probar todo lo posible hasta encontrar la solución o encontrar que no existe solución al problema. Para lograr esto, se separa la búsqueda en varias búsquedas parciales o subtareas, a la vez, estas subtareas suelen incluir más subtareas, por lo que se tratan de algoritmos recursivos.

Se llaman algoritmos de vuelta atrás porque en caso de no encontrar una solución al problema de una subtarea, se devuelve a la subtarea original y se intenta con distintos argumentos. Estos algoritmos se asemejan mucho al tratamiento de los grafos ya que cada subtarea podría tomarse como un grafo de la siguiente manera.

Figura 6. Visión de un algoritmo de vuelta atrás como un árbol.



Fuente: Backtracking [en linea]: Algoritmos de salto atrás. Barcelona: Algoritmia.net, 2007. [Consultado 02 de febrero de 2008]. Disponible en Internet: http://www.algoritmia.net/articles.php?key=backtracking

[23] WIRTH, Niklaus [en línea], Biografia. Zurich: Universidad de Zurich, 2008. [Consultado 28 de noviembre de 2007]. Disponible en Internet: http://www.cs.inf.ethz.ch/~wirth/

A menudo ocurre que el árbol o grafo que se genera es tan grande que encontrar una solución o encontrar la mejor solución entre varias posibles es computacionalmente muy costoso. En estos casos suelen aplicarse una serie de condiciones, de tal forma que se puedan podar algunas de las ramas, es decir, no recorrer ciertas subtareas. Esto es posible si llegado a un punto se puede demostrar que la solución que se obtendrá a partir de ese punto no será mejor que la mejor solución obtenida hasta el momento.

- Los algoritmos de vuelta atrás o backtracking tienen características similares y se pueden representar en el siguiente esquema propuesto por Wirth:

```
procedimiento ensayar (paso : TipoPaso)
  para cada candidato hacer
  |  seleccionar candidato
  |  if aceptable then
  |  begin
  |    anotar_candidato
  |    if solucion_incompleta then
  |      ensayar(paso_siguiente)
  |    else
  |      almacenar_solucion
  |    borrar_candidato
  |  end
  hasta que candidatos_agotados
fin procedimiento
```

8.6.3 Objetivos del caso de estudio.

- Demostrar la mejora de los tiempos de ejecución, de algoritmo de vuelta atrás (La vuelta del Caballo), entre un ambiente de Grid Computing y un ambiente de computación tradicional.

- Lograr el funcionamiento del algoritmo en ambiente de Grid.

- Realizar una comparación entre los tiempos de ejecución del algoritmo para los dos tipos de ambiente.

- Realizar la misma comparación anterior pero con mas recursos en el ambiente de Grid.

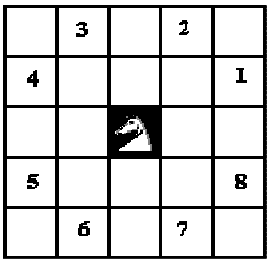
8.6.4 Solución Planteada. La vuelta del Caballo. Se dispone de un tablero rectangular, por ejemplo el tablero de ajedrez, y de un caballo, que se mueve según las reglas de este

juego. El objetivo es encontrar una manera de recorrer todo el tablero partiendo de una casilla determinada, de tal forma que el caballo pase una sola vez por cada casilla. Una variante es obligar al caballo a volver a la posición de partida en el último movimiento. Por último se estudiará como encontrar todas las soluciones posibles partiendo de una misma casilla.

Para resolver el problema hay que realizar todos los movimientos posibles hasta que ya no se pueda avanzar, en cuyo caso hay que dar marcha atrás, o bien hasta que se cubra el tablero. Además, es necesario determinar la organización de los datos para implementar el algoritmo.

¿Cómo se mueve un caballo? A continuación se muestran los 8 movimientos que puede hacer el caballo. Estos movimientos serán los candidatos.

Figura 8. Movimientos que puede hacer el caballo (nótese movimientos en L). El caballo esta en la posición (0,0)



Fuente: Backtracking. [en linea]. Algoritmos de vuelta atrás. Barcelona: Algoritmia.net y algo mas, 2007. [Consultado 02 de febrero de 2008] Disponible en Internet: http://www.algoritmia.net/articles.php?key=backtracking

Con las coordenadas en las que se encuentre el caballo y las ocho coordenadas relativas se determina el siguiente movimiento. Las coordenadas relativas se guardan en dos arreglos:

ejex = [2, 1, -1, -2, -2, -1, 1, 2]

ejey = [1, 2, 2, 1, -1, -2, -2, -1]

El tablero, del tamaño que sea, se representará mediante un array bidimensional de números enteros. A continuación se muestra un gráfico con un tablero de tamaño 5x5 con todo el recorrido partiendo de la esquina superior izquierda.

Figura 9. Solución al problema de un tablero de 5x5

1	16	11	6	3
10	5	2	17	12
15	22	19	4	7
20	9	24	13	18
23	14	21	8	25

En la figura 9 se muestra como se soluciona este problema, la primera posición es (1,1), la segunda es (3,2), la tercera es (5,1), la cuarta es (4,3) y así sucesivamente.

Se corre un algoritmo en C++ en el cual se presenta la solución a este problema en un ambiente tradicional de computación. Y se compara con computación en Grid de la siguiente manera.

- El siguiente es el algoritmo en lenguaje C:

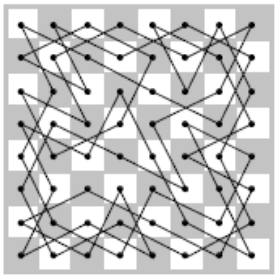
```
#include <stdio.h>

#define N 5
#define ncuad N*N

void mover(int tablero[][N], int i, int pos_x, int pos_y, int *q);

const int ejex[8] = { -1,-2,-2,-1, 1, 2, 2, 1 },
          ejey[8] = { -2,-1, 1, 2, 2, 1,-1,-2 };

int main(void)
{
  int tablero[N][N]; /* tablero del caballo. */
  int i,j,q;

   /* inicializa el tablero a cero */
  for (i = 0; i < N; i++)
   for (j = 0; j < N; j++)
    tablero[i][j] = 0;

   /* pone el primer movimiento */
  tablero[0][0] = 1;
  mover(tablero,2,0,0,&q);

  if (q) { /* hay solución: la muestra. */
    for (i = 0; i < N; i++) {
      for (j = 0; j < N; j++)
        printf("%3d ", tablero[i][j]);
      putchar('\n');
    }
  }
  else
    printf("\nNo existe solucion\n");
```

```
  return 0;
}

void mover(int tablero[][N],int i, int pos_x, int pos_y, int *q)
{
  int k, u, v;

  k = 0;
  *q = 0;
  do {
    u = pos_x + ejex[k]; v = pos_y + ejey[k]; /* seleccionar candidato */
    if (u >= 0 && u < N && v >= 0 && v < N) { /* esta dentro de los limites? */
      if (tablero[u][v] == 0) {  /* es valido? */
        tablero[u][v] = i;  /* anota el candidato */
        if (i < ncuad) {  /* llega al final del recorrido? */
          mover(tablero,i+1,u,v,q);
          if (!*q) tablero[u][v] = 0; /* borra el candidato */
        }
        else *q = 1; /* hay solución */
      }
    }
    k++;
  } while (!*q && k < 8);
}
```

Fuente: Backtracking. [en linea]: Algoritmos de vuelta atrás. Barcelona: Algoritmia.net y algo mas, 2007. [Consultado 02 de febrero de 2008] Disponible en Internet: http://www.algoritmia.net/articles.php?key=backtracking

Se puede observar en la siguiente figura los movimientos que hará el caballo pasando por todas las casillas del tablero al menos una vez en un tablero de 8x8.

Figura 10. Movimientos del caballo en un tablero de 8x8



Fuente: Backtracking. [en línea]. Barcelona: Algoritmia .net, 2007. [Consultado 02 de febrero de 2008] Disponible en Internet: http://www.algoritmia.net/articles.php?key=backtracking

8.6.5. Computación Tradicional. En aras que los resultados se acerquen lo más posible a la realidad, es decir en un ambiente tradicional y sin usar la filosofía de Grid; se utilizara el software que administra el Grid, pero para una sola máquina, lo cual es el equivalente a si se decidiera poner en cola los trabajos manualmente.

Tabla 5. Tiempos de Ejecución del algoritmo de La Vuelta del Caballo.

	Tiempo de ejecución (en segundos).	Tiempo Acumulado (en segundos)
N= 3	0.5	0.5
N=4	0.7	1.2
N=5	0.7	1.9
N=6	0.9	2.8
N=7	1	3.8
N=8	3	**6.8**

- Computación en Grid. La computación en Grid, ofrece entre muchas otras características la ejecución simultánea de tareas según los recursos que están disponibles en los diferentes nodos que lo componen. La ejecución del algoritmo de "la vuelta del caballo" se realizara de la siguiente manera. Se ejecutara el mismo algoritmo, con los mismos argumentos y se utilizara el mismo scheduler con el que se ejecuto en la computación tradicional.

Tabla 6. Tiempos de Ejecución del algoritmo de "La Vuelta del Caballo" en 2 máquinas trabajando en Grid. Nótese que el tiempo acumulado no es la suma de los tiempos de ejecución, ya que hay tareas que se ejecutan simultáneamente.

	Tiempo de ejecución.	Tiempo Acumulado
N= 3	0.5	0.5
N=4	0.7	0.7
N=5	0.7	1.2
N=6	0.9	1.6
N=7	1	2.2
N=8	3	**4.6**

Tabla 7. Tiempos de Ejecución del algoritmo de "La Vuelta del Caballo" en 3 máquinas trabajando en Grid.

	Tiempo de ejecución.	Tiempo Acumulado
N= 3	0.5	0.5
N=4	0.7	0.7
N=5	0.7	0.7
N=6	0.9	1.4
N=7	1	1.7

N=8	3	**3.7**

8.7 SELECCIÓN DEL CASO DE ESTUDIO - QUICKSORT.

8.7.1. Introducción. QuickSort es un algoritmo de ordenamiento con un tiempo de ejecución de orden $O(n^2)$ en el peor de los casos para un conjunto de datos n. A pesar de este tiempo de ejecución, este algoritmo es muy utilizado porque en su caso promedio[24] su tiempo de ejecución es de orden O(n log n).

Esta es probablemente la técnica más rápida conocida. Fue desarrollada por Hoare[25] en 1960. El algoritmo original es recursivo, pero se utilizan versiones iterativas para mejorar su rendimiento (los algoritmos recursivos son en general más lentos que los iterativos, y consumen más recursos).

8.7.2. Algoritmos "Divide y Vencerás"[26]. El termino divide y vencerás hace alusión a dividir un problema muy grande en varios subproblemas y así sucesivamente hasta que los subproblemas son de solución sencilla, la solución al problema inicial se construye con la unión de la respuesta de estos subproblemas.

En las ciencias de la computación, el término **divide y vencerás** (**DYV**) hace referencia a uno de los más importantes paradigmas de diseño algorítmico. El método está basado en la resolución recursiva de un problema dividiéndolo en dos o más subproblemas de igual tipo o similar. El proceso continúa hasta que éstos llegan a ser lo suficientemente sencillos como para que se resuelvan directamente. Al final, las soluciones a cada uno de los subproblemas se combinan para dar una solución al problema original. Esta técnica es la base de los algoritmos eficientes para casi cualquier tipo de problema como, por ejemplo, algoritmos de ordenamiento (quicksort, mergesort, entre muchos otros) y la transformada discreta de Fourier.

- A continuación se presenta un pseudocódigo de un algoritmo Divide y vencerás:

```
AlgoritmoDYV(p: TipoProblema): TipoSolucion
    IF esCasoBase(p)
       return resuelve(p)
```

[24] Orden de ejecución [en línea]: Algoritmos de ordenación. Cataluña: Universidad de Cataluña, 2008. [Consultado 13 de octubre de 2007]. Disponible en Internet: www.lsi.upc.edu/~duch/home/duch/alg_ord.pdf

[25] C.A.R Hoare. [en línea]: Biografia del Autor. New York: Microsoft Research, 2008. [Consultado 25 de febrero de 2008]. Disponible en Internet : http://research.microsoft.com/users/thoare/

[26] Tipos de Algoritmos [en línea]: Algoritmo Divide y Vencerás. Florida: Wikipedia Foundation, 2008. [Consultado 15 de febrero de 2008]. Disponible en Internet: http://es.wikipedia.org/wiki/Divide_y_vencer%C3%A1s

```
    ELSE
       subproblemas: ARRAY OF TipoProblema
       subproblemas = divideEnSubproblemas(p)
       soluciones_parciales: ARRAY OF TipoSolucion

       FOR EACH sp IN subproblemas
          soluciones_parciales.push_back(AlgoritmoDYV(sp))

       return mezcla(soluciones_parciales)
```

Fuente: Tipos de Algoritmos [en línea]: Algoritmo Divide y Vencerás. Florida: Wikipedia Foundation, 2008. [Consultado 15 de febrero de 2008]. Disponible en Internet: http://es.wikipedia.org/wiki/Divide_y_vencer%C3%A1s

8.7.3. Objetivos del Caso de Estudio

- Demostrar la mejora de los tiempos de ejecución para un conjunto de datos muy grande (entre 1'000.000 de datos de 30 dígitos y 4'000.000 de datos), de un algoritmo "Divide y Vencerás" en este caso el QuickSort, entre un ambiente de Grid Computing y un ambiente de computación tradicional.

- Lograr el funcionamiento del algoritmo divide y vencerás en ambiente de Grid.

- Realizar una comparación entre los tiempos de ejecución del algoritmo para los dos tipos de ambiente.

- Realizar la misma comparación anterior pero con mas recursos en el ambiente de Grid.

- Permitir formular una hipótesis la cual permita tener un estimado (formula) sobre los tiempos de ejecución del algoritmo QuickSort en un ambiente en Grid.

8.7.4. Solución planteada. Para la solución planteada se realiza una descripción breve del QuickSort.

El algoritmo fundamental es el siguiente:

- Elegir un elemento de la lista de elementos a ordenar, que se llama **pivote**.

- Resituar los demás elementos de la lista a cada lado del pivote, de manera que a un lado queden todos los menores que él, y al otro los mayores. En este momento, el pivote ocupa exactamente el lugar que le corresponderá en la lista ordenada.

- La lista queda separada en dos sublistas, una formada por los elementos a la izquierda del pivote, y otra por los elementos a su derecha.

- Repetir este proceso de forma recursiva para cada sublista mientras éstas contengan más de un elemento. Una vez terminado este proceso todos los elementos estarán ordenados.

- Como se puede suponer, la eficiencia del algoritmo depende de la posición en la que termine el pivote elegido.

- En el mejor caso (caso promedio), el pivote termina en el centro de la lista, dividiéndola en dos sublistas de igual tamaño. En este caso, el orden de complejidad del algoritmo es $O(n \cdot \log n)$.

- En el peor caso, el pivote termina en un extremo de la lista. El orden de complejidad del algoritmo es entonces de $O(n^2)$. El peor caso dependerá de la implementación del algoritmo, aunque habitualmente ocurre en listas que se encuentran ordenadas, o casi ordenadas.

- En el caso promedio, el orden es $O(n \cdot \log n)$.

- No es extraño, pues, que la mayoría de optimizaciones que se aplican al algoritmo se centren en la elección del pivote.

- A continuación se presenta el código en Java a implementar bajo un ambiente tradicional y un ambiente en Grid.

```
/**
 * @(#)GeneraString.java
 * @author Jairo Fernando Ortiz Zapata
 * @version 1.00 2008/5/29
 */
public class GeneraString {
   private String datos [];
   private int cantidad_datos;
   private final int cantidad_digitos = 70;
   private String abecedario="abcdefghijklmnopqrstuvwxyz";
   public GeneraString(int cantidad_datos) {
      datos = new String [cantidad_datos];
      this.cantidad_datos = cantidad_datos;
    }
    public void llenar_arreglo (){
      String palabra="" ;
      int numero;
```

```
      for (int i=0;i<cantidad_datos;i++){
            for (int j=0;j<cantidad_digitos;j++){
                  numero = (int)(Math.random()*25);
            //    System.out.println("numero"+numero);
                  palabra += abecedario.charAt(numero);
            }
            datos [i]=palabra;
            palabra="";
      }
   }
   public void qsort (String arreglo[], int izq, int der) {
      int inf = izq;//limite inferior
      int sup = der;//limite superior
      int pivote;//elemento pivote (elemento referencia)
      String temp ="";//temporal que permite el intercambio de datos
      pivote = (inf+der)/2;
      do{
            //si hay un elemento mayor al pivote lo deja indicado
            while( (arreglo[inf].compareTo(arreglo[pivote])<0) && inf<=der){
                  inf++;
            }
            //si hay un elemento menor al pivote lo deja indicado
            while( (arreglo[sup].compareTo(arreglo[pivote])>0) && sup>=izq){
                  sup--;
            }

            if( inf <= sup ){
                  temp = arreglo[inf];
                  arreglo[inf] = arreglo[sup];
                  arreglo[sup] = temp;
                  /*aumenta inferior y disminuye el superior para seguir con
                   *el siguiente dato*/
                  inf++;
                  sup--;
            }
      }while(inf<=sup);//controla que los limites no se traslapen
      if( izq < sup ){//si se puede dividir por la parte izquierda del arreglo
            qsort( arreglo, izq, sup );}//llamado recursivo
      if( inf < der ){//si se puede dividir por la parte derecha del arreglo
            qsort( arreglo, inf, der );}//llamado recursivo
   }//qsort
   public void imprimir (){
      for (int i=0;i<cantidad_datos;i++){
            System.out.println(""+datos[i]);
      }
   }
   public static void main (String ar[]){
      GeneraString obj = new GeneraString (Integer.parseInt(ar[0]));
      obj.llenar_arreglo();
      System.out.println("\nDatos Ordenados\n\n\n");
      obj.qsort(obj.datos,0,obj.datos.length-1);
      obj.imprimir();
   }
}
```

8.7.5 Computación Tradicional. En aras que los resultados se acerquen lo más posible a la realidad, es decir en un ambiente tradicional y sin usar la filosofía de Grid; se utilizara el

software que administra el Grid, pero para una sola máquina, lo cual es el equivalente a si se decidiera poner en cola los trabajos manualmente.

Las pruebas realizadas al algoritmo se constituyen de la siguiente manera: ordenar conjuntos de 15’950.000 (N) datos de longitud 70 cada palabra (ej: nenendfgddfgdfgdfgdfgdgdfjfuqwgbcnmlkljklxdfgaasdfgndasdasdasdfdsghfgh). El problema se subdividido en 22 grupos de de la siguiente manera: n>=200.000 donde n se incrementa de 50.000 en 50.000 (ej: un grupo tiene n=200.000 datos, otro tiene n=250.000 datos, etc.).

La tabla mediante la cual se tomaron los tiempos de ejecución fue la siguiente:
Hora inicial: 01:06:50.

Tabla 8. Tiempos de ejecución del algoritmo Quicksort para un total de datos de 15’950.000 bajo un ambiente de computación tradicional.

n	Tiempo en mm:ss
200.000	1:09
250.000	1:23
300.000	1:43
350.000	1:58
400.000	2:04
450.000	2:18
500.000	2:37
550.000	3:03
600.000	3:17
650.000	3:29
700.000	3:46
750.000	4:08
800.000	4:20
850.000	4:36
900.000	4:59
950.000	5:20
1’000.000	5:37
1’050.000	5:53
1’100.000	6:19
1’150.000	6:56
1’200.000	7:27
1’250.000	9:51

El tiempo total de la prueba es de 91 minutos y 13 segundos bajo el ambiente tradicional, para una hora final de 02:38:03 (hh:mm:ss).

8.7.6 Computación en Grid. La computación en Grid, ofrece entre muchas otras características la ejecución simultánea de tareas según los recursos que están disponibles en los diferentes nodos que lo componen. La ejecución del algoritmo QuickSort se realizara de la siguiente manera. Se ejecutara el mismo algoritmo, con la misma cantidad de datos dividida de la misma forma que en la computación tradicional para que sea una comparación valida.

La tabla mediante la cual se tomaron los tiempos de ejecución fue la siguiente:
Hora inicial: 03:06:50.

Tabla 9. Tiempos de ejecución del algoritmo Quicksort para un total de datos de 15’950.000 bajo un ambiente de computación en Grid.

n	Tiempo en mm:ss
200.000	1:08
250.000	1:24
300.000	1:43
350.000	1:56
400.000	2:04
450.000	2:20
500.000	2:39
550.000	3:03
600.000	3:17
650.000	3:29
700.000	3:43
750.000	4:06
800.000	4:20
850.000	4:36
900.000	4:59
950.000	5:20
1’000.000	5:37
1’050.000	5:56
1’100.000	6:19
1’150.000	6:56
1’200.000	7:28
1’250.000	9:50

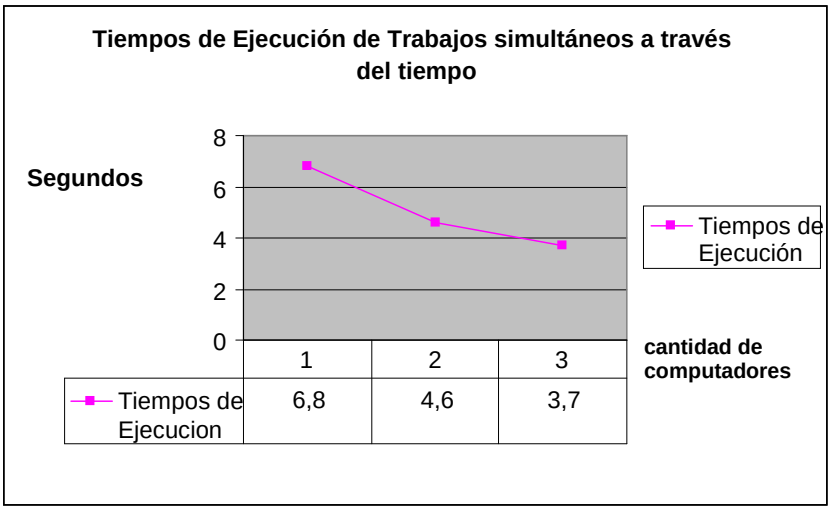
El tiempo total de la prueba es de 43 minutos y 48 segundos bajo el ambiente tradicional, para una hora final de 03:50:38 (hh:mm:ss). Este tiempo final se presenta ya que hay tareas que se ejecutan de forma simultanea en cuantos recursos el demonio condor_collector encuentre disponibles.

8.8 ANALISIS DE RESULTADOS DE PRUEBAS

El análisis de las pruebas que se realizaron mediante el caso de estudio, los cuales se realizaron tanto en ambiente de Grid como bajo un ambiente tradicional. El análisis que se presenta a continuación está dividido en secciones. Las pruebas realizadas en este proyecto están adecuadas según los recursos disponibles y se hacen proporcionales para que se visualicen de forma veraz los resultados obtenidos.

8.8.1 Mejoramiento de los tiempos de ejecución. Según las pruebas realizadas en los laboratorios de la Universidad Autónoma de Occidente, se observó que a medida que se van aumentando los recursos computacionales los tiempos de ejecución final de todo un trabajo con diferentes argumentos va disminuyendo a una tasa poco proporcional de la siguiente manera según la siguiente figura:

Figura 11. Tiempos de Ejecución de Trabajos simultáneos a través del tiempo, para el algoritmo de vuelta del caballo.



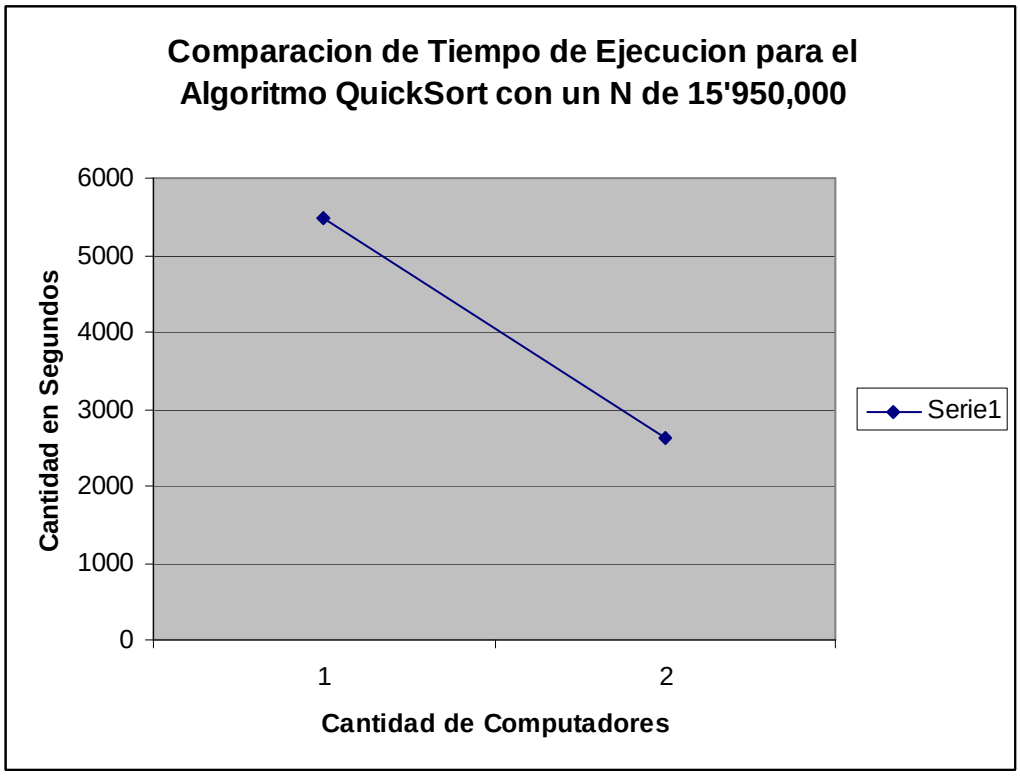
Se puede observar que el tiempo no disminuye linealmente, ya que uno de los factores es que hay que negociar recursos computacionales disponibles en el Grid. Además la asignación de recursos también influye.

También hay que tener en cuenta que Condor asigna recursos de acuerdo a la disponibilidad de los mismos, se podría mejorar el tiempo de ejecución si los algoritmos de configuración de Cóndor se modificaran de tal manera que asignara recursos de acuerdo con el algoritmo que se va a ejecutar, ya que si se ejecuta un algoritmo de orden exponencial en una máquina obsoleta, el tiempo de ejecución final va a ser mucho mayor que si se ejecutara en una máquina con más capacidad de procesamiento.

En el caso de estudio 2 (ejecución del algoritmo de Quicksort para un conjunto de datos del orden de los 15 millones, en donde cada dato se compone de 70 caracteres) se puede observar fundamentalmente que el tiempo de ejecución disminuyo en más de la mitad y paso de ser de 91 minutos a 43 minutos mediante la ejecución en dos maquinas de manera simultanea, esto quiere decir, que cuanto mejor este configurada el software que administra uno de los clusters que hace parte del Grid, mejor va a ser la capacidad de respuesta de este software en la asignación de recursos con las tareas solicitadas.

A continuación se presenta un grafico donde se muestra de que manera va disminuyendo el tiempo de ejecución a medida que se aumentan los recursos disponibles en el Grid.

Figura 12. Comparación de Tiempo de Ejecución para el Algoritmo QuickSort con un N de 15'950,000.



Vale la pena aclarar que el tiempo de ejecución T siempre es mayor que 0 sin importar la cantidad de recursos disponibles en el Grid.

8.8.2 Trabajo simultáneo entre trabajos. Como se expuso anteriormente, los trabajos lanzados a ejecución al Grid (cluster) se lanzaron para que se ejecutaran de manera simultánea. El algoritmo ejecutado se ejecuto en su totalidad en una máquina y con diferentes argumentos. Este tipo de computación permite que el proceso de asignación de

recursos cuando se presenta disponibilidad, disminuya el tiempo de ejecución final del trabajo que se encolo en Cóndor.

En el caso de estudio de la vuelta del caballo, tanto como en la ejecución del algoritmo QuickSort para un volumen muy grande de datos, se demostró que a medida que aumentan los recursos en el Grid y por ende en el cluster, se pueden ejecutar trabajos de manera simultánea, lo cual garantiza que los tiempos de ejecución sean menores y se garantice que siempre habrá recursos computacionales explotados al máximo de su capacidad. Esta característica está restringida por la cantidad de memoria y procesador disponible, ya que si las máquinas que hacen parte del Grid están ocupadas, el maestro debe estar en la capacidad de gestionar recursos para ejecutar la tarea.

Estos resultados de las pruebas conlleva a decir que se puede implementar algoritmos paralelos mediante librerías de PVM (Parallel Virtual Machine) y "MPI(Message Passing Interface)"[27] los cuales tienen beneficios como la ejecución de múltiples tareas en un solo procesador, dividir un algoritmo para que se ejecute en varias máquinas de forma paralela.

8.8.3 Análisis posterior a los resultados. Según los resultados obtenidos se observó que el cluster no solo mejora los tiempos de ejecución de un algoritmo, sino que su poder de procesamiento de datos es muy grande, el uso adecuado de esta herramienta y de esta nueva filosofía permitiría que tareas que se demoren días pueda llegar a un tiempo de ejecución de horas según la configuración del Grid y de la cantidad de recursos disponibles.

Se podría determinar que la computación en Grid se comporta mejor para ejecuciones de algoritmos que requieran gran capacidad de procesamiento, ya que según los casos de estudios mencionados y desarrollados en este trabajo, la mejoría se ve reflejada de una manera más clara en la ejecución del algoritmo QuickSort para un conjunto de datos muy elevado que para un conjunto de datos pequeño como el señalado en el caso de estudio (la vuelta del caballo).

Se concluye que entre más computadores hagan parte del Grid mejor van a ser los tiempos de ejecución de los algoritmos que requieran gran procesamiento de datos como los que son de orden exponencial. También se visualiza que mediante la ejecución de algoritmos de forma simultanea se mejora de forma casi lineal los tiempos de ejecución de dichos algoritmos.

[27] Manual de condor, Op. Cit., Disponible en Internet : http://www.cs.wisc.edu/condor/manual/v6.8/ref.html

8.8.4 Hipótesis formulada. Si aumentando de uno a tres computadores conectados mediante un HUB se mejoro el tiempo de ejecución final de un algoritmo de 6.8 a 3.7, ¿cómo seria si se colocaran recursos del orden de 1000 computadores conectados en Grid? ¿De qué manera influye el tráfico de red y la negociación de recursos a través de un Grid de más de una organización virtual?

9. CONCLUSIONES

Aunque existen muchas y diversas iniciativas para la instalación automatizada de clusters, la implementación de un cluster con un software ya conocido como lo es Condor, requiere de mucho más que un soporte realizado por un manual detallado del software, requiere un análisis de la arquitectura que se implementará, así como los casos de estudio para las pruebas que se realicen. Además, por ser una investigación de una nueva tecnología, requiere tiempo y planificación de las actividades a realizar con mucho tiempo de holgura.

Uno de los cambios más importantes en la investigación de Grid Computing es el cambio de la filosofía de compartir recursos con otros dominios y con otras organizaciones (Virtualización), ya que esta tecnología permite el establecer recursos disponibles para un conjunto de tareas que han sido solicitadas. Estos recursos computacionales pueden están ubicados geográficamente en distintos lugares y administrados por diferentes dominios y software, y en dichos dominios, por políticas de seguridad, no se comparten recursos con otros dominios.

Definitivamente si en las organizaciones se realizaran exhaustivas investigaciones en el tema de Grid Computing, el mundo se daría cuenta que verdaderamente la unión hace la fuerza y algoritmos que antes no eran posibles ser solucionados de manera satisfactoria, ahora si seria posible solucionarlos.

En el presente proyecto se dibujo un cluster lo más parecido posible a un Grid para conocer de primera mano que efectivamente la computación en Grid ofrece una capacidad de computación de alto desempeño tanto para organizaciones de bajo presupuesto, como para grandes organizaciones (p.ej. Universidades).

A pesar que el cluster implementado en este proyecto, fue con solo 3 computadores, se alcanzó a dimensionar que ni el software a ejecutar en el Grid, ni los resultados tiene que estar centralizado en una máquina maestra, es tan solo necesario que el software que administre el Grid sea centralizado y configurado en todas las máquinas que hacen parte de él.

Mediante la comparación de la computación tradicional y la computación en Grid, se concluyó que a medida que los recursos computacionales aumenten, mejora el tiempo de ejecución del problema que se ha puesto en la cola del Grid, ya que la negociación de recursos que hacen parte de éste hacen que cada vez que encuentra recursos disponibles se van asignando tareas, cuyo tiempo de ejecución final va a estar dado por el peor de los casos que será el tiempo que se tarde la máquina más lenta en terminar un trabajo.

El algoritmo implementado en el caso de estudio 1 y puesto en marcha en el cluster, se dimensionó de acuerdo con los recursos disponibles, ya que no se obtendrían resultados relevantes si se corre un algoritmo con un conjunto de datos muy grande en pocas máquinas.

Para el caso de estudio en el cual se visualiza la ejecución de un algoritmo con un conjunto de datos del orden de los 15 millones de datos, y la ejecución del algoritmo de vuelta del caballo con un conjunto de datos acorde a los recursos disponibles, se observo claramente que tanto para un conjunto de datos pequeño como para un procesamiento de gran volumen de datos, la computación en Grid ofrece mejores tiempos de ejecución, siempre y cuando la configuración del software que lo administre esté acorde con la arquitectura de los computadores, de la red y de los clusters en general.

10. RECOMENDACIONES

Para una futura investigación en Grid Computing, se debe tener en cuenta los conceptos de las personas relacionadas en el tema, ya que el soporte encontrado en Internet no es del todo fiable, ya que por ser una tecnología nueva, no todas las personas saben el paradigma y la forma de pensar que hay que adquirir cuando se habla de Grid Computing.

Se recomienda seguir la investigación en el tema de Grid Computing porque actualmente se están conformando comunidades a nivel nacional como Grid Colombia, liderada por la Universidad de los Andes, y comunidades a nivel internacional como RENATA y el proyecto EELA, en donde la Universidad Autonoma de Occidente haría un gran aporte en el avance de esta nueva tecnología y avance en los diferentes tópicos que en las que ésta se aplica.

En proyectos de investigación de implementación de nuevas tecnologías, se deben disponer de mas recursos computacionales debido a que se dimensionarían problemas de mayor envergadura como es el tratamiento de imágenes, soluciones medicas, avances en el descubrimiento de soluciones a problemáticas mundiales, etc. Además se debe tener en cuenta que en este tipo de proyectos se debe contar con una disponibilidad del 100% del tiempo para la investigación ya que ésta requiere de mucha documentación.

BIBLIOGRAFIA

Butterfly.net [en línea]: Powering Next-Generation Gaming with On-Demand Computing. San Francisco: ZDNET, 2008. [Consultado 15 de enero de 2008]. Disponible en Internet: http://whitepapers.zdnet.com/casestudy.aspx?dtid=3&scid=197&docid=81560

Caso de Estudio de Mercadeo [en línea]: Caso de estudio de Marketing. New York: Secretos en Red, 2008 [Consultado 21 de abril de 2008]. Disponible en Internet: http://secretosenred.com/articles/4641/1/CASO-ESTUDIO-DE-MARKETING-DIVISION-FASTAXI-DE-

CNN. Ciencia y tecnologia [en líena]: Grid Computing to aid breast cancer research. Atlanta: CNN, 2001. [Consultado 19 de octubre de 2008]. Disponible en internet: http://archives.cnn.com/2001/TECH/industry/11/29/grid.cancer.research.idg/

El salto del caballo [en línea]. Barcelona, Algoritmia. Net, 2008. [Consultado 19 de abril de 2008]. Disponible en Internet: http://www.algoritmia.net/problems.php?id=27

FOSTER, Ian. The Globus Proyect [en linea]: A Status Report. Chicago: Argonne National Laboratory, 2007. [Consultado 25 de octubre de 2007]. Disponible en Internet: http://citeseer.ist.psu.edu/cache/papers/cs/14468/ftp:zSzzSzftp.globus.orgzSzpubzSzglobuszSzpaperszSzglobus-hcw98.pdf/foster98globus.pdf

FOSTER, Ian. What is the Grid? [en linea]: A three point checklist. Chicago: University of Chicago, 2002. [Consultado 20 de octubre de 2007]. Disponible en internet: http://www.mellanox.com/pdf/whitepapers/GridTech_WhitePaper_final.pdf

Manual de Globus [en línea]: Breve Guia de Instalacion de Globus Toolkit 3.2.1 – Draft. New York: Globus, 2008. [Consultado 15 de diciembre de 2007]. Disponible en Internet: http://www.globus.org/mail_archive/java/2005/05/pdf00000.pdf

Grid Computing with Globus [en línea]: Introduction to Grid Computing with Globus. New York: IBM, 2008. [Consultado el 2 de febrero de 2008]. Disponible en Internet: http://www.redbooks.ibm.com/redbooks/pdfs/sg246895.pdf

Manual de Condor [en linea]: Manual de usuario. Wisconsin: University of Wisconsin-Madison, 2008. [Consultado 19 de diciembre de 2007]. Disponible en Internet: http://www.cs.wisc.edu/condor/manual/v6.8/ref.html

Soluciones en Grid [en línea]: IBM y Butterfly crean un innovador entorno de juegos en red para Playstation®2. Madrid: IBM, 2008. [Consultado 27 de febrero de 2008]. Disponible en Internet: http://www-5.ibm.com/es/press/notas/2003/febrero/butterfly.html

SOTOMAYOR, Borja. Computacion en Grid [en linea]: Introducción a la computación Grid. Chicago: Universidad de Chicago, 2004. [Consultado 23 de octubre de 2007]. Disponible en Internet: http://people.cs.uchicago.edu/~borja/lectures/IntroduccionGrid.pdf

NASH, Miranda. Oracle 10g [en linea]: Infraestrucure for Grid Computing. New York: Oracle Corporation, 2003. [Consultado 15 de octubre de 2007]. Disponible en internet: http://www.mellanox.com/pdf/whitepapers/GridTech_WhitePaper_final.pdf

PARBERRY, Ian. Problems on Algorithms [en linea]: Divide and Conquer. Texas: Universidad del Norte de Texas, 2002. [Consultado 28 de Abril de 2008]. Disponible en Internet: http://www.eng.unt.edu/ian/books/free/poa.pdf

Q. Peng D.P. Schissel M. Thompson I. Foster M. Greenwald D. McCune K. Keahey T. Fredian. The Anatomy Of The Grid [en linea]: Enabling Scalable Virtual Organizations. Chicago: Universidad de Chicago, 2002. [Consultado 17 de octubre de 2007]. Disponible en internet: http://www.globus.org/alliance/publications/papers/anatomy.pdf

SCHOPF, Jennifer M. Grids [en linea]: Top Ten Question. North Carolina: Northwestern University, 2003. [Consultado 24 de octubre de 2007]. Disponible en Internet: http://www.globus.org/alliance/publications/papers/topten.final.pdf

ANEXOS

Anexo A. Configuración de Condor

- Configuración de Condor: Tomado directamente del manual de administración de Condor

Newer Unix Installation Procedure

The Perl script condor configure installs Condor. Command-line arguments specify all needed information to this script. The script can be executed multiple times, to modify or further set the configuration. condor configure has been tested using Perl 5.003. Use this or a more recent version of Perl.

After download, all the files are in a compressed, tar format. They need to be untarred, as

```
tar xzf completename.tar.gz
```

After untarring, the directory will have the Perl script condor configure, as well as a second tar file called release.tar. condor configure works on release.tar.
condor configure is completely command-line driven; it is not interactive. Several commandline arguments are always needed with condor configure. The argument

```
--install=/path/to/release.tar
```

specifies the path to the Condor release tarball. The argument

```
--install-dir=directory
```

specifies the path to the install directory. The argument

```
--local-dir=directory
```

specifies the path to the local directory.

The **–type** option to condor configure specifies one or more of the roles that a machine may take on within the Condor pool: central manager, submit or execute. These options are given in a comma separated list. So, if a machine is both a submit and execute machine, the proper command-line option is

```
--type=manager,execute
```

Configure Condor on the central manager machine first. If Condor will run as root in this pool (Item 3 above), run condor configure as root, and it will install and set the file permissions correctly.
On the central manager machine, run condor configure as follows.

```
% condor_configure --install=r.tar --install-dir=~condor \
--local-dir=/scratch/condor --type=manager
```

The central manager can also be a submit point or and execute machine, but this is only recommended for very small pools. If this is the case, the **–type** option changes to manager,executeor manager,submit or manager,submit,execute.

After the central manager is installed, the execute and submit machines should then be configured. Decisions about whether to run Condor as root should be consistent throughout the pool. Foreach machine in the pool, run

```
% condor_configure --install=r.tar --install-dir=~condor \
--local-dir=/scratch/condor --type=execute,submit
```